ΑΛΕΞΑΝΔΡΕΙΟ Τ.Ε.Ι. ΘΕΣΣΑΛΟΝΙΚΗΣ
ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Πτυχιακή Εργασία

«Διασύνδεση βάσεων δεδομένων εταιρείας αποδελτίωσης εντύπων»

Της φοιτήτριας
Εβελίνα Ριμπάποβα
Αρ. Μητρώου: 032214

Επιβλέπων καθηγητής
Δέρβος Δημήτρης

Θεσσαλονίκη 2011

Θα ήθελα να ευχαριστήσω θερμά τους
γονείς μου για την στήριξη και τον αγώνα
που έκαναν για εμένα.
Επίσης, θα ήθελα να ευχαριστήσω τον
Θεοδόση για την υποστήριξη που μου
προσέφερε.

Περίληψη

Σκοπός της παρούσας πτυχιακής είναι η δημιουργία ενός προγράμματος σε JAVA το οποίο θα καλύψει συγκεκριμένες ανάγκες εταιρείας αποδελτίωσης εντύπων. Συγκεκριμένα, θα ενημερώνει την MS SQL βάση δεδομένων λαμβάνοντας υπόψη τις αλλαγές στα δεδομένα που έχουν πραγματοποιηθεί στην υπάρχουσα Oracle βάση, θα εισάγει στην MS SQL βάση τα δεδομένα που θα αντλεί από XML αρχεία τα οποία δέχεται το πρόγραμμα από μία διεύθυνση εισόδου και θα μετατρέπει αρχεία PDF μορφής σε αρχεία DJVU μορφής με τη βοήθεια εξωτερικών προγραμμάτων. Για την επίτευξη της ζητούμενης λειτουργικότητας και απόδοσης, η εφαρμογή είναι multithreaded, κρατάει log αρχεία, έχει αρχείο παραμετροποίησης και είναι κατά το μέγιστο δυνατό αυτοματοποιημένη.

Η σχεδίαση και δημιουργία της νέας MS SQL βάσης περιλαμβάνεται στην πτυχιακή.

Abstract

The purpose of this dissertation is to create an application in JAVA that will cover the specific needs of a press clipping company. Specifically, it will update the MS SQL database according to changes in the data that exist in an oracle database, it will insert in the MS SQL database data coming from XML files which the program will acquire from a given input path and it will convert files from PDF format to DjVu with the help of external software.To achieve functionality and performance, the application will have to be multithreaded, to keep log files, to have a configuration file and to be as automated as possible.

The design and the creation of the new MS SQL database is included in the dissertation.

Πίνακας περιεχομένων

Ευρετήριο εικόνων

Ευρετήριο Πινάκων

Ευρετήριο Κώδικα

Εισαγωγή

Ο ρόλος που συχνά ένας εργαζόμενος τμήματος μηχανοργάνωσης καλείται να εκπληρώσει, είναι να στηρίξει και να καλύψει τις ειδικευμένες ανάγκες του τρέχοντος συστήματος της εταιρείας, όπου εργάζεται. Αυτό συνήθως σημαίνει είτε δημιουργία νέου λογισμικού, είτε προσθήκη νέων δυνατοτήτων, είτε παραμετροποίηση του υπάρχοντος συστήματος έτσι ώστε να εξυπηρετούνται καλύτερα οι ανάγκες της εταιρείας. Αυτό συμβαίνει επειδή η αγορά ή παραγγελία λογισμικού με βάση τις ακριβείς ανάγκες της εταιρείας είναι κάτι το οποίο κοστίζει πολύ και είναι δύσκολο μια επιχείρηση να επενδύσει σε κάτι τέτοιο, ειδικά αν οι ανάγκες της είναι πολλές και μεγάλες. Η δεύτερη εναλλακτική, η οποία είναι αυτή που κατά κανόνα εφαρμόζεται, είναι η αγορά έτοιμου λογισμικού το οποίο θα καλύψει στο μεγαλύτερο ποσοστό τις ανάγκες τις επιχείρησης αλλά δεν θα μπορέσει να ανταποκριθεί στις ιδιαιτερότητές της.

Η παρούσα πτυχιακή εργασία στοχεύει στην ανάπτυξη εφαρμογής η οποία θα γεφυρώσει το κενό που υπάρχει μεταξύ του υποσυστήματος παραγωγής, των αποκομμάτων εταιρείας αποδελτίωσης και το υποσύστημα δημοσίευσης αυτών στο Internet. Ουσιαστικά θα είναι ο συνδετικός κρίκος για να μπορούν τα δύο αυτά υποσυστήματα να λειτουργούν συγχρονισμένα. Οι προδιαγραφές τις εφαρμογής και η περιγραφή του περιβάλλοντος στο οποίο αυτή λειτουργεί είναι πραγματικές.

Οι ευαίσθητες για την εταιρεία πληροφορίες είτε έχουν τροποποιηθεί, είτε έχουν παραληφθεί.

Η εταιρεία αποδελτίωσης εντύπων έχει την κυριότητα των πνευματικών δικαιωμάτων της τεχνογνωσίας, της μεθοδολογίας αποδελτίωσης, του κώδικα όπως και της βάσης δεδομένων που αναπτύχθηκε στην παρούσα πτυχιακή. Οποιαδήποτε μη εξουσιοδοτημένη χρήση, εφαρμογή, ανατύπωση, αντιγραφή μέρους ή ενότητας του κώδικα και ότι άλλο ισχύει με βάση την νομοθεσία, διώκεται ποινικά.

Μέσα από την παρούσα πτυχιακή εργασία θα μας δοθεί η ευκαιρία να σας ξεναγήσουμε στον όχι και τόσο γνωστό, αλλά σίγουρα ενδιαφέροντα κόσμο της αποδελτίωσης. Θα αναφέρουμε τον ορισμό και την διαδικασία με την

οποία αυτή υλοποιείται στις μέρες μας. Θα δούμε τα πραγματικά προβλήματα και τις πραγματικές ανάγκες που προέκυψαν για να πετύχουμε ένα σύστημα λειτουργικό και αυτοματοποιημένο έτσι ώστε τα αποκόμματα να είναι διαθέσιμα στους πελάτες. Για την επίτευξη αυτού του στόχου χρειάστηκε να αναπτυχθεί νέο λογισμικό σε JAVA και να επανασχεδιαστεί η υπάρχουσα βάση δεδομένων. Οι προδιαγραφές και οι υλοποίηση περιγράφονται στα επόμενα κεφάλαια.

Κεφάλαιο 1: Αποδελτίωση εντύπων

1.1 Εισαγωγή

Η εφαρμογή την οποία θα αναφέρουμε και θα αναλύσουμε σε επόμενα κεφάλαια είναι άρρηκτα συνδεδεμένη με το περιβάλλον στο οποίο «ζει» και «αναπνέει» και αυτό δεν είναι άλλο από την εταιρεία αποδελτίωσης εντύπων. Έτσι, θα πρέπει να αναφερθούμε και να εξηγήσουμε το τι είναι η αποδελτίωση εντύπων, ποια είναι και πως λειτουργεί η αντίστοιχη παραγωγική διαδικασία, ποιό είναι το προϊόν της και πώς η εφαρμογή η οποία αναπτύχθηκε στο πλαίσιο της παρούσας πτυχιακής εργασίας ενσωματώνεται στο συγκεκριμένο περιβάλλον.

1.2 Τι είναι η αποδελτίωση τύπου

Με τον όρο «τύπο» ή «έντυπα» αναφερόμαστε σε πάσης φύσεως εφημερίδες και περιοδικά τα οποία κυκλοφορούν στην αγορά.

Όπως πολύ σωστά αναφέρεται στην Wikipedia, *«Αποδελτίωση Τύπου είναι η διαδικασία με την οποία επιλεγμένα άρθρα με ενδιαφέρουσα θεματολογία, καταχωρούνται "ως έχουν" και αρχειοθετούνται ώστε να είναι προσβάσιμα και επεξεργάσιμα με την πρώτη αναζήτηση. Μπορεί να είναι και υπηρεσία κατά την οποία ο πελάτης αναθέτει σε ειδικευμένη εταιρεία την παρακολούθηση συγκεκριμένης θεματολογίας και την οποία του την παραδίδει σε επεξεργασμένη μορφή, διατηρώντας την αρχική μορφή του εντύπου.».* Στην περίπτωσή μας η εταιρεία αποδελτίωσης αναλαμβάνει την παρακολούθηση του έντυπου τύπου για πληροφορίες που ενδιαφέρουν τους πελάτες της και τις παραδίδει αυτούσιες, με τη μορφή αποκόμματος, από το έντυπο μέσο, προσθέτοντας πληροφορίες για το έντυπο και το απόκομμα αυτό καθ' αυτό.

Οποιαδήποτε εμπορική επιχείρηση, για παράδειγμα, μπορεί να ενδιαφέρεται για ανακοινώσεις χρηματοδοτήσεων μέσω προκηρύξεων. Επιπρόσθετα μπορεί να ενδιαφέρεται για τον ανταγωνισμό της, και συγκεκριμένα, για τις προσφορές που διαφημίζει στον έντυπο τύπο και για τις διαφημίσεις του, αυτές καθ' αυτές. Σίγουρα μια εταιρεία θα ενδιαφέρονταν να

ξέρει σε ποια άρθρα κάποιος έχει γράψει κάτι, το οποίο είτε απλά αναφέρεται στην εταιρεία, είτε σχολιάζεται κάτι για αυτήν, θετικά ή αρνητικά, αξιολογώντας έτσι το αν το άρθρο αυτό μετρά ως θετική ή αρνητική διαφήμιση. Έρευνες και δημοσκοπήσεις που γίνονται είναι επίσης σημείο ενδιαφέροντος, όπως και πολλά άλλα αναλόγως με το είδος της επιχείρησης. Όλες αυτές οι ανάγκες είναι υπαρκτές σχεδόν για την κάθε εταιρεία. Δεν μπορούν όμως να ικανοποιήσουν αυτές τους τις ανάγκες μόνες τους και αυτό γιατί, πέρα από οικονομικούς λόγους, δεν έχουν την απαραίτητη τεχνογνωσία και το κατάλληλο ανθρώπινο δυναμικό. Έτσι προέκυψε και η ανάγκη ύπαρξης των γραφείων αποδελτίωσης τύπου.

1.3 Σύντομη ιστορική αναδρομή

Η αποδελτίωση έκανε πρώτα την εμφάνισή της με τη μορφή ενός ή περισσοτέρων ατόμων τα οποία προσλαμβάνονταν από μια εταιρεία η οποία ενδιαφέρονταν για το τι γράφει ο τύπος για αυτήν. Τα άτομα αυτά, τα οποία εργάζονταν εντός της εταιρείας και μόνον για αυτήν, διάβαζαν, σάρωναν στα γρήγορα όλες τις σημαντικότερες εφημερίδες και έκοβαν ή κύκλωναν τα άρθρα τα οποία παρουσίαζαν ενδιαφέρον. Όπως γίνεται αντιληπτό, αρχικά όλη η διεργασία της αποδελτίωσης διεκπεραιώνονταν και εξαρτιόταν πλήρως από τον ανθρώπινο παράγοντα. Φυσικά, δεν υπήρχε ακρίβεια και δεν μπορούσε να καλυφθεί όλος ο τύπος.

Με τα χρόνια οι υπολογιστές αναπτύχθηκαν και η προσφορά τους στην αποδελτίωση άλλαξε ριζικά το σκηνικό. Δημιουργήθηκαν εταιρίες αποδελτίωσης και χρόνο με το χρόνο οι δυνατότητες των υπολογιστών βελτίωσαν την ποιότητα, ποσότητα και ταχύτητα εξαγωγής των ζητουμένων πληροφοριών. Ο ανθρώπινος παράγοντας δεν παραλείφθηκε αλλά «περιορίστηκε» στο παραγωγικό κομμάτι του λογικού ελέγχου και τις κάλυψης αναγκών που ακόμα και μέχρι σήμερα αδυνατούν οι υπολογιστές να καλύψουν. Φτάνοντας στο σήμερα, η διαδικασία είναι κατά 95% αυτοματοποιημένη, όμως το υπόλοιπο 5% που αντιστοιχεί στον ανθρώπινο παράγοντα είναι ίσως και το πιο κρίσιμο, το οποίο κάνει και την διαφορά.

1.4 Σύνοψη

Η υπηρεσία της αποδελτίωσης εντύπων προσφέρει την παρακολούθηση του τύπου για θέματα που αφορούν τους πελάτες της. Η συμβολή των υπολογιστών για την λειτουργία της υπηρεσίας αυτής τα τελευταία χρόνια ήταν τεράστια, αυτοματοποιώντας σχεδόν ολόκληρη την διαδικασία παραγωγής των αποκομμάτων και αυξάνοντας την ταχύτητα των διεργασιών της.

Κεφάλαιο 2: Περιβάλλον και αντικείμενο της πτυχιακής εργασίας

2.1 Εισαγωγή

Σε αυτό το κεφάλαιο θα περιγράψουμε το περιβάλλον στο οποίο θα λειτουργεί η εφαρμογή της παρούσας πτυχιακής εργασίας.

Το περιβάλλον αυτό αποτελείται από δύο υποσυστήματα. Το υποσύστημα αποδελτίωσης, στο οποίο δημιουργούνται τα αποκόμματα των πελατών, και το υποσύστημα δημοσίευσης, όπου καταχωρούνται τα αποκόμματα που έχουν παραχθεί, έτσι ώστε να είναι προσβάσιμα από τους πελάτες της εταιρείας.

Τέλος, θα περιγράψουμε τον ρόλο που θα κατέχει η εφαρμογής της παρούσας πτυχιακής εργασίας εντός αυτού του περιβάλλοντος.

2.2 Υποσύστημα αποδελτίωσης της εταιρίας αποδελτίωσης.

Το υποσύστημα αποδελτίωσης είναι το σημαντικότερο κομμάτι της αποδελτίωσης και είναι αυτό το οποίο παράγει τα αποκόμματα. Στη συνέχεια θα περιγράψουμε την σειρά των διεργασιών που ακολουθείται για την αποδελτίωση.

Το πρώτο στάδιο περιλαμβάνει προμήθεια και στη συνέχεια σάρωση όλου του έντυπου τύπου της Ελληνικής αγοράς. Αφού πλέον τα έντυπα υπάρχουν σε ηλεκτρονική μορφή μπορούν έπειτα να επεξεργαστούν από τις υπόλοιπες εφαρμογές του υποσυστήματος (βλ. *Εικόνα 1: Ψηφιοποίηση εντύπων σε μορφή εικόνας*).

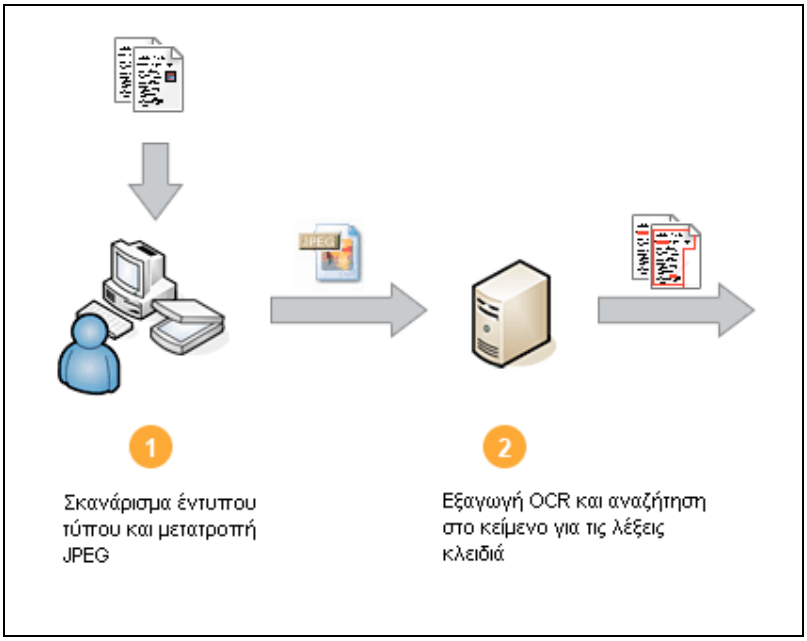
Εικόνα 1: *Ψηφιοποίηση εντύπων σε μορφή εικόνας*

Στο δεύτερο στάδιο οι σαρωμένες σελίδες περνάνε από το επονομαζόμενο OCR (Optical Character Recognition - Οπτική Αναγνώριση Χαρακτήρων). Συνοψίζοντας τον ορισμό που δίνεται στην Wikipedia, η οπτική αναγνώριση χαρακτήρων είναι η τεχνολογία που επιτρέπει την αναγνώριση και εξαγωγή του κειμένου από ηλεκτρονικά έγγραφα, αντικαθιστώντας έτσι τη χρονοβόρα διαδικασία της αντιγραφής με πληκτρολόγηση.

Το OCR είναι ο μόνος τρόπος να εξάγεις και να μετατρέψεις σε αναγνωρίσιμο από υπολογιστές, το κείμενο μιας σαρωμένης εικόνας. Αυτή η μέθοδος είναι το «κλειδί» και ίσως το σημαντικότερο κομμάτι όλης της διαδικασίας. Έχοντας το κείμενο της κάθε σαρωμένης σελίδας σε ηλεκτρονική μορφή δίνεται η δυνατότητα αναζήτησης σε αυτήν. Έτσι, με αυτοματοποιημένο τρόπο μέσω υπολογιστών, πραγματοποιείται η αναζήτηση σε θέματα ενδιαφέροντος των πελατών της εταιρείας αποδελτίωσης μέσω απλών ή σύνθετων λέξεων κλειδιών. Επιπρόσθετα, υπάρχει η δυνατότητα ο πελάτης να αναζητήσει οτιδήποτε θελήσει από τα αποκόμματα που έχει λάβει όπως και το να αντιγράψει και να επεξεργαστεί εύκολα το κείμενο των αποκομμάτων του, κάτι που δεν θα ήταν εφικτό διαφορετικά.

Αυτό που ακολουθεί είναι η αναζήτηση των λέξεων κλειδιών που ενδιαφέρουν τους πελάτες, αυτόματα από το σύστημα. Η κάθε λέξη κλειδί που έχει βρεθεί επισημαίνεται στην σελίδα με κόκκινο χρώμα. Επίσης, η αρχή και

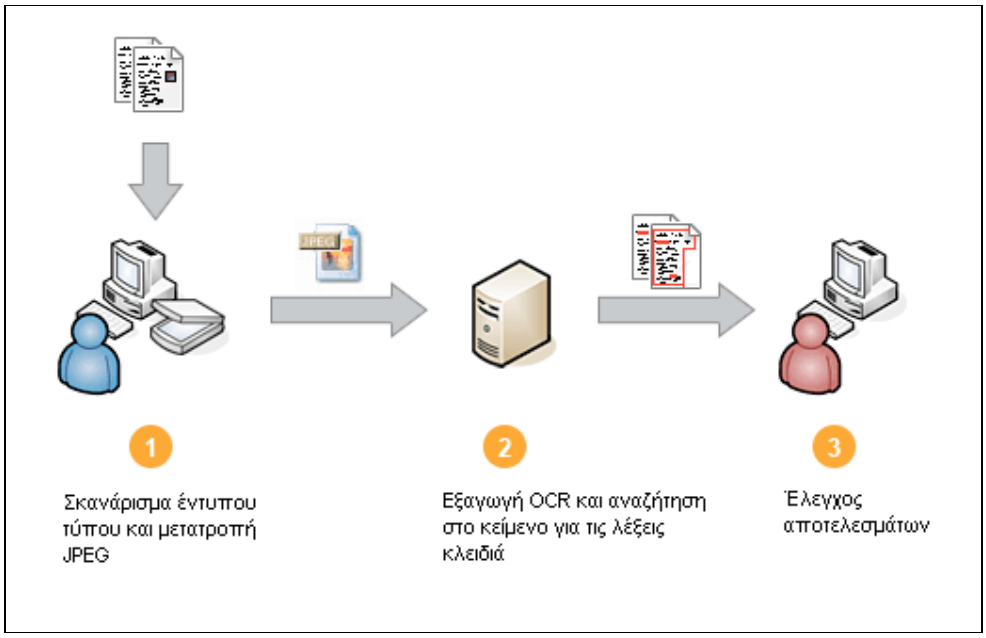
το τέλος του κάθε άρθρου αναγνωρίζεται και μαρκάρεται από το σύστημα. Η οριοθετημένη αυτή περιοχή της σελίδας του εντύπου που αποτελείται από ένα και μόνο άρθρο είναι το βασικό στοιχείο του αποκόμματος (clip) (βλ. *Εικόνα 2: Αναγνώριση κειμένου για την κάθε σελίδα, αναζήτηση λέξεων κλειδών σε αυτές και οριοθέτηση αποκομμάτων*).



Εικόνα 2: *Αναγνώριση κειμένου για την κάθε σελίδα, αναζήτηση λέξεων κλειδών σε αυτές και οριοθέτηση αποκομμάτων*

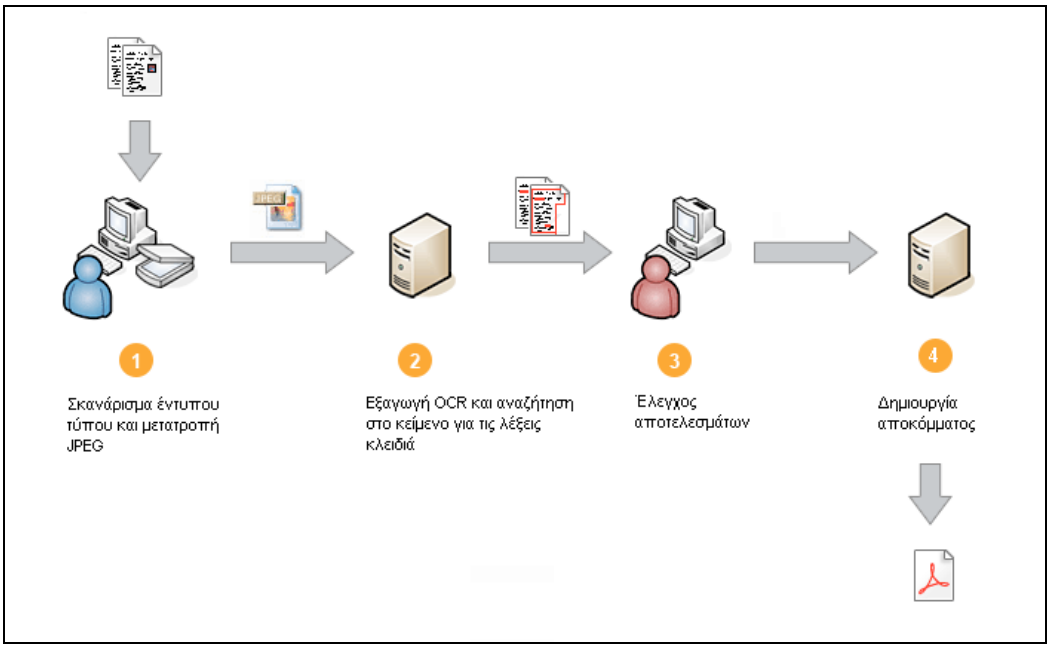
Στη συνέχεια τα άρθρα ελέγχονται από ανθρώπινο παράγοντα ως προς το αν έχουν οριοθετηθεί σωστά από το σύστημα και αν οι μαρκαρισμένες με κόκκινο λέξεις κλειδιά αφορούν όντως τον πελάτη.

Για παράδειγμα, αν ο πελάτης ενδιαφέρεται για την εταιρεία Apple, το σύστημα θα φέρει αποτελέσματα όχι μόνο για την εταιρεία Apple αλλά και για οτιδήποτε αναφέρεται στα μήλα (το φρούτο) σε αγγλικό κείμενο, το οποίο πιθανότατα δεν θα ενδιαφέρει το πελάτη. Ο ανθρώπινος παράγοντας είναι αυτός που ταυτοποιεί τα αποτελέσματα στο επίπεδο που η τεχνολογία δεν μπορεί να ανταποκριθεί μέχρι σήμερα. Ευτυχώς, τα ποσοστά λάθους του συστήματος είναι πολύ μικρά (βλ.*Εικόνα 3: Έλεγχος αποτελεσμάτων συστήματος από ανθρώπινο παράγοντα*).



Εικόνα 3: *Έλεγχος αποτελεσμάτων συστήματος από ανθρώπινο παράγοντα*

Στο τελευταίο στάδιο, από το κάθε άρθρο που έχει ελεγχθεί δημιουργείται αυτόματα αρχείο PDF το οποίο εμπλουτίζεται, προσθέτοντας μια κεφαλίδα στο καθένα, η οποία περιέχει χρήσιμες πληροφορίες για το άρθρο (βλ. *Εικόνα 4: Δημιουργία αρχείου PDF για το κάθε άρθρο* και *Εικόνα 5: Δείγμα αποκόμματος αποδελτίωσης*).



Εικόνα 4: *Δημιουργία αρχείου PDF για το κάθε άρθρο*

Το απόκομμα (βλ. *Εικόνα 5: Δείγμα αποκόμματος αποδελτίωσης*) αποτελείται από μια λευκή σελίδα όπου στο κέντρο της έχει τοποθετηθεί το άρθρο που ενδιαφέρει τον πελάτη. Στην κεφαλίδα, στην πάνω αριστερή γωνία, αναγράφονται στοιχεία του εντύπου στο οποίο βρέθηκε το άρθρο όπως και πληροφορίες σχετικά με το μέγεθος του άρθρου.

ΚΕΡΔΟΣ
Είδος: Εφημερίδα / Κύρια / Οικονομική / Ημερήσια
Ημερομηνία: Παρασκευή, 14-01-2011
Σελίδα: 56
Μέγεθος: 78 cm ²
Μέση κυκλοφορία: 560
Επικοινωνία εντύπου: (210) 6747.880
Λέξη κλειδί: ΠΛΗΡΟΦΟΡΙΚΗ

ΖΗΤΟΥΝ ΕΡΓΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Πάγια αιτήματα του **Συνδέσμου Επιχειρήσεων Πληροφορικής και Επικοινωνιών Ελλάδας (ΣΕΠΕ)** τέθηκαν χθες σε συνάντηση εκπροσώπων του ΔΣ του ΣΕΠΕ με τον υπουργό Περιφερειακής Ανάπτυξης και Ανταγωνιστικότητας κ. **Μιχάλη Χρυσοχοΐδη**. Στη συνάντηση, όπου παρευρέθηκαν, μεταξύ άλλων, ο πρόεδρος του ΣΕΠΕ κ. **Παντελής Τζωρτζάκης,** ο ΣΕΠΕ μετέφερε τη θέση ότι οι τεχνολογίες πληροφορικής και επικοινωνιών αποτελούν όχι απλώς ενδεδειγμένη, αλλά τη μοναδική απάντηση στα καίρια προβλήματα της χώρας αλλά και στον σκόπελο της οικονομικής κρίσης. Κατά τη διάρκεια της συνάντησης, συζητήθηκαν θέματα που αφορούν στην ανάγκη επιτάχυνσης των δράσεων του Επιχειρησιακού Προγράμματος «Ψηφιακή Σύγκλιση». Εμφαση δόθηκε και στην προώθηση έργων τόσο με τη μέθοδο συμπράξεων δημοσίου ιδιωτικού τομέα (ΣΔΙΤ) όσο και με τη μέθοδο εκχώρησης επιχειρησιακών λειτουργιών (Business Process Outsourcing).

Εικόνα 5: *Δείγμα αποκόμματος αποδελτίωσης*

Το σύνολο των διεργασιών που περιγράφονται παραπάνω πραγματοποιείται από μία σουίτα προγραμμάτων. Όλα τα προγράμματα

επικοινωνούν με συγκεκριμένη βάση δεδομένων. Το κάθε πρόγραμμα εξειδικεύεται σε κάποια διεργασία.

Στην κοινή, για όλα τα προγράμματα, βάση δεδομένων καταχωρούνται όλες οι απαιτούμενες πληροφορίες που χρειάζονται οι εφαρμογές για να λειτουργήσουν. Συγκεκριμένα, είναι καταχωρημένοι όλοι οι πελάτες της εταιρείας με όλα τα απαραίτητα στοιχεία τους (τηλέφωνο, email, διεύθυνση, υπεύθυνοι κλπ), οι λέξεις κλειδιά που τους ενδιαφέρουν (οι οποίες ουσιαστικά αποτελούν το θεματολόγιο του πελάτη), τον τρόπο με τον οποίο θέλει να παραλαμβάνει την αποδελτίωσή του κ.α.

Η εύρεση των πληροφοριών που ενδιαφέρουν τον πελάτη, επί του συνόλου των εντύπων, γίνεται με βάση τις λέξεις κλειδιά που έχει καθορίσει.

Επίσης ο πελάτης λαμβάνει τα αποκόμματά του μέσο ηλεκτρονικής δημοσίευσης τους στο Internet, στα οποία έχει πρόσβαση από τον δικτυακό τόπο της εταιρείας μέσω λογαριασμού που έχει δημιουργηθεί αποκλειστικά για αυτόν.

2.3 Υποσύστημα δημοσίευσης των αποκομμάτων

2.3.1 Ιστότοπος

Αυτό το οποίο πληρώνει και λαμβάνει ο πελάτης ως προϊόν από την υπηρεσία της αποδελτίωσης εντύπων είναι τα αποκόμματα, τα οποία έχουν δημιουργηθεί από το παραγωγικό σύστημα το οποίο περιγράφεται στην προηγούμενη ενότητα. Έχει ήδη αναφερθεί ότι τα αποκόμματα δημοσιοποιούνται στον εταιρικό ιστότοπο. Ο πελάτης, συνδέεται με τον λογαριασμό του σε αυτόν τον ιστότοπο και παρακολουθεί την αποδελτίωση που έχει παραχθεί για αυτόν.

Επιπρόσθετα, στον εταιρικό ιστότοπο παρέχεται η δυνατότητα να βλέπει ο πελάτης το κάθε ένα απόκομμά του είτε σε μορφοποίηση PDF, είτε σε μορφοποίηση DjVu, επιλέγοντας αυτό το οποίο τον εξυπηρετεί καλύτερα.

2.3.2 Αποκόμματα σε μορφοποίηση PDF/DjVu

Πριν συνεχίσουμε θα γίνει μια περιγραφή των προτερημάτων και των μειονεκτημάτων της μορφοποίησης PDF έναντι της μορφοποίησης DJVU.

Η μορφοποίηση DjVu έχει σχεδιαστεί για να επιτυγχάνει μικρότερο μέγεθος αρχείου για αρχεία τα οποία αποτελούνται από κείμενο και εικόνες. Αυτό εφαρμόζεται επακριβώς στα σαρωμένα αρχεία από τον έντυπο τύπο. Το μεγάλο προτέρημα της μορφοποίησης DjVu είναι το γεγονός ότι χρησιμοποιεί τεχνολογίες οι οποίες επιτρέπουν το αρχείο να έχει μέγεθος κατά πολύ μικρότερο με το αντίστοιχο PDF και έτσι να είναι πιο εύκολη η διάθεσή του μέσω του διαδικτύου. Για την ακρίβεια, διαθέτει έως και 5 φορές μικρότερο μέγεθος αρχείου σε σχέση με το PDF, και ταυτόχρονα, σχεδόν την ίδια ποιότητα. Έτσι, χρησιμοποιώντας μορφοποίηση DJVU, επιτυγχάνεται ταχύτερο κατέβασμα και χαμηλότερη επιβάρυνση του διαθέσιμου bandwidth.

Από την άλλη πλευρά, μεγάλο μειονέκτημα συνιστά το γεγονός ότι η συγκεκριμένη τεχνολογία δεν έχει καθιερωθεί (ακόμα) ως πρότυπο στο Internet. Για να αναγνωρίσει ο υπολογιστής αυτή τη συγκεκριμένη μορφοποίηση χρειάζεται η εγκατάσταση ειδικού plug-in. Η εγκατάσταση του plug-in είναι κάτι το οποίο δεν μπορεί να εύκολα να πραγματοποιηθεί σε πολλούς πελάτες λόγω περιορισμών από την μηχανοργάνωση εντός της εταιρείας τους.

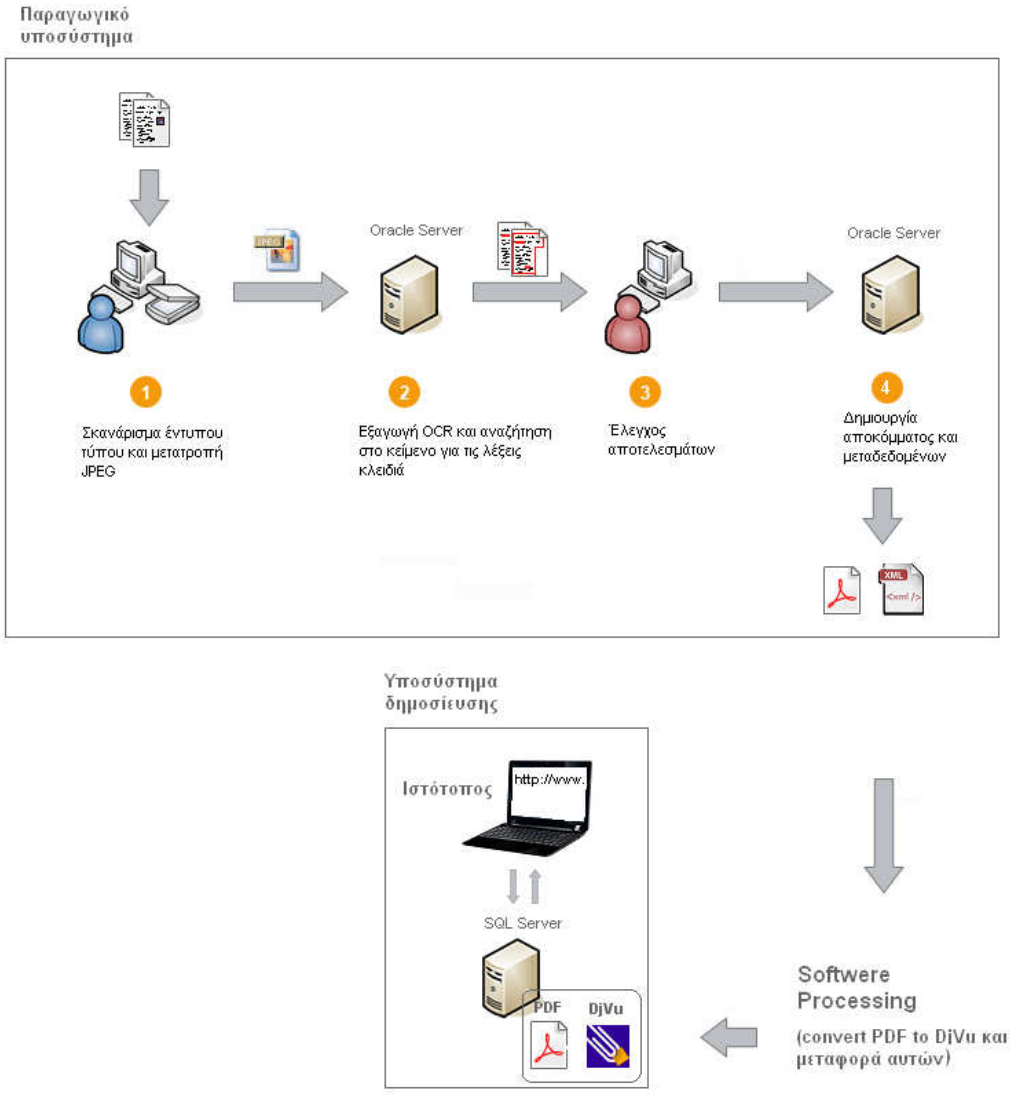
Αντιθέτως, η μορφοποίηση PDF, έχει καθιερωθεί και έχει διαδοθεί παγκοσμίως. Ως συνέπεια αυτού, όλοι οι υπολογιστές σήμερα μπορούν να αναγνωρίσουν τον συγκεκριμένο τύπο αρχείου.

Έτσι, το PDF μπορεί να λύσει το πρόβλημα με τους πελάτες οι οποίοι δεν μπορούν να εγκαταστήσουν το DjVu plug-in. Ευτυχώς, αυτοί οι πελάτες είναι λίγοι. Οι πλειοψηφία των πελατών χρησιμοποιεί το DjVu πετυχαίνοντας έτσι καλύτερες ταχύτητες και άρα καλύτερη ανταπόκριση στην πλοήγησή τους.

Όπως είδαμε στο υποσύστημα της παραγωγικής διαδικασίας, τα αποκόμματα δημιουργούνται σε μορφή PDF. Πώς λοιπόν ο πελάτης έχει την δυνατότητα να δει τα αποκόμματά του σε μορφοποίηση DjVu; Χρησιμοποιώντας το κατάλληλο λογισμικό το οποίο μετατρέπει τη μορφοποίηση PDF σε DjVu. Η δημιουργία του αρχείου «on the fly», δηλαδή

την στιγμή που το ζητάει ο πελάτης, δεν είναι δυνατή επειδή ο χρόνος που απαιτείται για αυτήν την μετατροπή είναι μεγάλος.

Αξιολογώντας τα προτερήματα και τα μειονεκτήματα αποφασίστηκε να δημιουργείται το DjVu αρχείο από το PDF και να αποθηκεύονται και τα δύο ταυτόχρονα. Δηλαδή το κάθε απόκομμα θα υπάρχει δύο φορές, μία φορά σε DjVu και μία φορά σε μορφοποίηση PDF. Μειονέκτημα αυτής της λύσης είναι ότι υπάρχει έτσι μεγαλύτερη (έως και πενταπλάσια) ανάγκη για αποθηκευτικό χώρο. Αυτό όμως που κοστίζει εν έτει 2011 περισσότερο, είναι το bandwidth. Όπως φαίνεται στην *Εικόνα 6: Παρεμβολή JAVA εφαρμογής μεταξύ του παραγωγικού υποσυστήματος και του υποσυστήματος δημοσίευσης με σκοπό να δημιουργεί νέα αρχεία DJVU από το υπάρχον PDF. Τέλος, μεταφέρει σε κατάλληλη τοποθεσία τα αρχεία*, χρειάζεται ένα λογισμικό το οποίο θα «διαβάζει» το PDF αρχείο του αποκόμματος και θα το μετατρέπεται σε αρχείο μορφοποίησης DjVu. Φυσικά, η διαδικασία της μετατροπής θα γίνεται με χρήση εξωτερικών προγραμμάτων. Τα παραγόμενα DjVu και PDF αρχεία θα αποθηκεύονται στο υποσύστημα δημοσίευσης, σε σημείο όπου ο ιστότοπος θα έχει πρόσβαση μέσω ενός δεύτερου DBMS.



Εικόνα 6: *Παρεμβολή JAVA εφαρμογής μεταξύ του παραγωγικού υποσυστήματος και του υποσυστήματος δημοσίευσης με σκοπό να δημιουργεί νέα αρχεία DJVU από το υπάρχον PDF. Τέλος, μεταφέρει σε κατάλληλη τοποθεσία τα αρχεία*

2.4 Διασύνδεση του παραγωγικού υποσυστήματος με το υποσύστημα δημοσίευσης των αποκομμάτων.

Όπως έχει ήδη αναφερθεί, το σύστημα της αποδελτίωσης αποτελείται από το υποσύστημα της παραγωγής και το υποσύστημα δημοσίευσης. Το κάθε ένα υποσύστημα έχει το δικό του DBMS. Το Oracle DBMS υποστηρίζει μόνο την παραγωγή αποκομμάτων, δηλαδή «ανήκει» στο παραγωγικό υποσύστημα. Το MSSQL DBMS υποστηρίζει το υποσύστημα δημοσίευσης και «ανήκει» σε αυτό. Μέχρι στιγμής υφίσταται έλλειψη ύπαρξης διασύνδεσης. Οι πληροφορίες των αποκομμάτων όπως και τα αποκόμματα αυτά καθ' αυτά πρέπει να μεταφερθούν στο υποσύστημα δημοσίευσης για να μπορεί ο πελάτης να παρακολουθεί την αποδελτίωσή του.

Πριν αναφερθεί ο τρόπος μεταφοράς της πληροφορίας, χρειάζεται να γίνει κατανοητός ο λόγος για τον οποίο έχει προκύψει η ανάγκη ύπαρξης δύο DBMS, εφόσον το πιο λογικό θα ήταν ο ιστότοπος να αντλεί τις απαιτούμενες πληροφορίες απευθείας από το παραγωγικό DBMS.

Η ύπαρξη δύο DBMS, αντί του ενός, δικαιολογείται λόγω των διαφορετικών αναγκών και της διαφορετικής λογικής μεταξύ του παραγωγικού υποσυστήματος της εταιρείας και των αναγκών του υποσυστήματος δημοσίευσης (ιστότοπος εταιρείας).

Το παραγωγικό υποσύστημα είναι εξαιρετικά σύνθετο σε αντίθεση με αυτό της δημοσίευσης, το οποίο είναι αρκετά πιο απλό.

Συγκρίνοντας τα δύο DBMS, αυτό της δημοσίευσης των αποκομμάτων αποτελεί ένα υποσύνολο του παραγωγικού υποσυστήματος. Στα δύο DBMS ουσιαστικά υπάρχουν κοινά δεδομένα τα οποία όμως είναι οργανωμένα και αποθηκευμένα με διαφορετική λογική.

Στο DBMS του παραγωγικού υποσυστήματος (Oracle), υπάρχει το πελατολόγιο, τα στατιστικά για τις αποστολές των αποκομμάτων (πληροφορίες για το ποια αποκόμματα μπόρεσαν ή δεν μπόρεσαν να σταλούν στους πελάτες), οι αναλυτικές περιγραφές των λέξεων κλειδιών, οι συντεταγμένες της τοποθεσίας της λέξης κλειδί στο απόκομμα και άλλα πολλά τα οποία δεν ενδιαφέρουν το υποσύστημα δημοσίευσης. Από την άλλη πλευρά, το DBMS της δημοσίευσης αποκομμάτων περιέχει πληροφορίες για

την εμφάνιση του ιστότοπου, λογαριασμούς πρόσβασης πελατών και πολλές ακόμα πληροφορίες οι οποίες δεν ενδιαφέρουν το DBMS του παραγωγικού υποσυστήματος.

Άλλος πολύ σημαντικός παράγοντας που οδήγησε στην ύπαρξη δύο ξεχωριστών DBMS ήταν η ασφάλεια από πιθανές επιθέσεις μιας και το παραγωγικό υποσύστημα είναι αποκομμένο από το διαδίκτυο.

Έχοντας δύο DBMS, προκύπτει η ανάγκη μεταφοράς των απαραίτητων και μόνο πληροφοριών από το παραγωγικό στο υποσύστημα δημοσίευσης. Αυτές οι πληροφορίες είναι τριών ειδών:

Α) Πληροφορίες που αφορούν τα αποκόμματα της αποδελτίωσης, όπως τον τίτλο του αποκόμματος, τον αρθρογράφο, την ημερομηνία δημοσίευσης, την ημερομηνία αποδελτίωσης, τον κωδικό του πελάτη τον οποίο αφορά, την λέξη κλειδί κ.α.

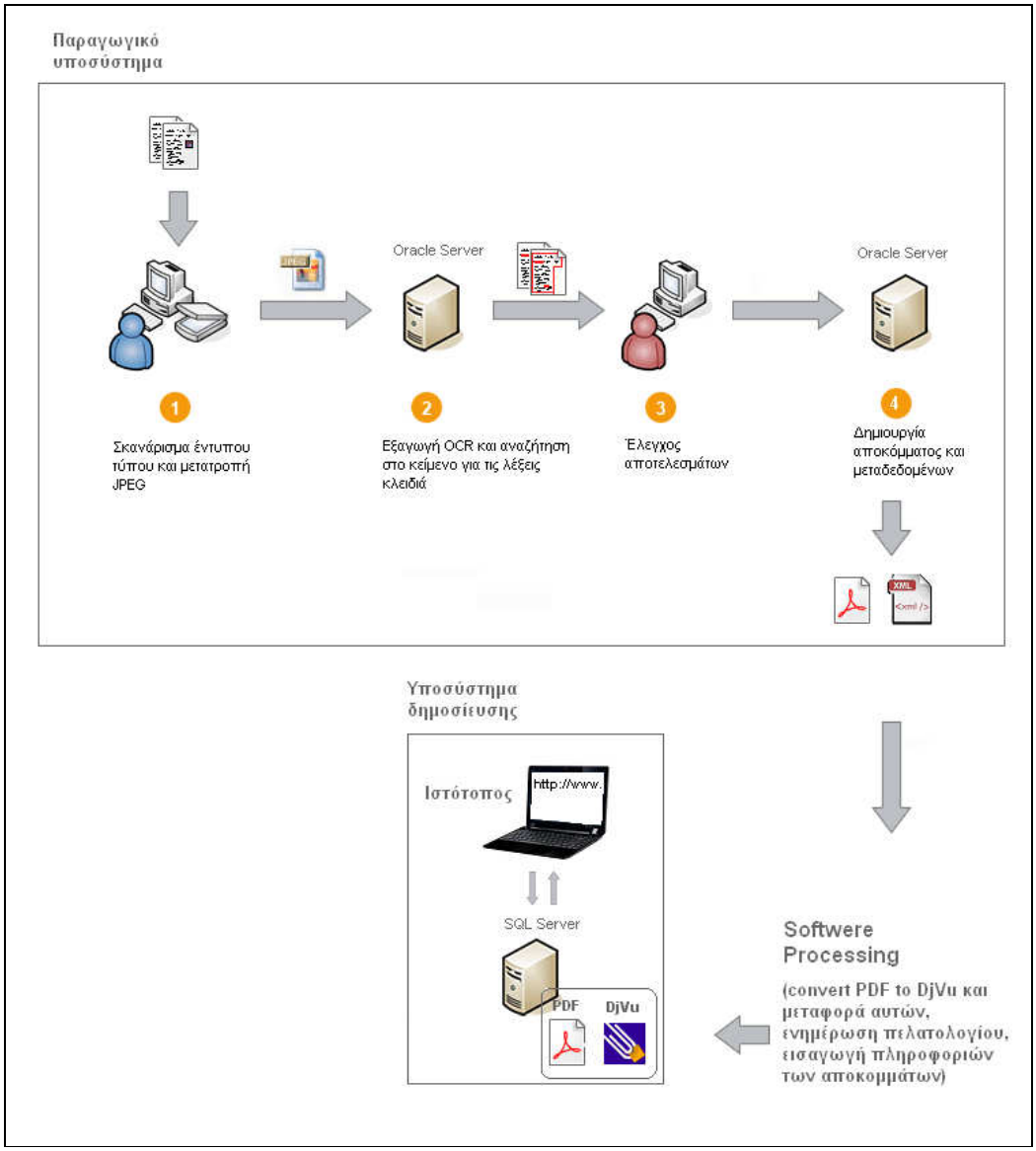
   Αυτές οι πληροφορίες μπορούν εύκολα να οργανωθούν σε αρχεία XML μιας και η δυνατότητα αυτή δίνεται από το παραγωγικό υποσύστημα. Έτσι κατά το τελευταίο στάδιο της παραγωγής του αποκόμματος σε PDF, δημιουργείται ταυτόχρονα και ένα αρχείο XML το οποίο περιέχει όλες τις απαραίτητες πληροφορίες του αποκόμματος και του αντίστοιχου πελάτη (βλ. *Εικόνα 7: Διασύνδεση των δύο υποσυστημάτων μέσω JAVA εφαρμογής*).

Β) Πληροφορίες που αφορούν τους πελάτες. Οι νέοι πελάτες που εισάγονται στο παραγωγικό υποσύστημα πρέπει αυτόματα να δημιουργούνται και στο υποσύστημα δημοσίευσης. Όμοια, οι πελάτες που διακόπτουν την συνεργασία τους θα πρέπει να διαγράφονται από το υποσύστημα δημοσίευσης. Με τον ίδιο τρόπο πρέπει να συγχρονίζονται όλες οι αλλαγές στις πληροφορίες και υπηρεσίες του πελάτη μεταξύ των δύο υποσυστημάτων.

   Ο συγχρονισμός αυτός των δεδομένων μπορεί να γίνει εύκολα από λογισμικό το οποίο θα ελέγχει και θα πραγματοποιεί τις αλλαγές σε ημερήσια βάση.

Γ) Το απόκομμα αυτό καθ' αυτό. Το PDF όπως και το αντίστοιχο DJVU αρχείο πρέπει να μεταφερθούν στο υποσύστημα δημοσίευσης για να έχουν πρόσβαση σε αυτά οι πελάτες.

Για την κάλυψη όλων των πιο πάνω αναγκών χρειαζόταν ένα πρόγραμμα το οποίο θα διασυνδέει τα δύο αυτά υποσυστήματα, όπως φαίνεται και στην *Εικόνα 7: Διασύνδεση των δύο υποσυστημάτων μέσω JAVA εφαρμογής*.



Εικόνα 7: *Διασύνδεση των δύο υποσυστημάτων μέσω JAVA εφαρμογής*

Αντικείμενο της συγκεκριμένης πτυχιακής εργασίας συνιστά η γεφύρωση του χάσματος μεταξύ αυτών των δύο DBMS έτσι ώστε να είναι εφικτή η ύπαρξη των ανεξάρτητων βάσεων, εξυπηρετώντας η κάθε μία τον δικό της σκοπό, ενώ ταυτόχρονα συγχρονίζονται τα απαραίτητα δεδομένα.

2.5 Σύνοψη

Όπως έχει ήδη αναφερθεί, αποδελτίωση είναι η παρακολούθηση του τύπου για συγκεκριμένα θέματα. Λόγω της πολυπλοκότητας του DBMS στην οποία λειτουργούν οι εφαρμογές του συστήματος της αποδελτίωσης προκύπτει η ανάγκη δημιουργίας ενός νέου DBMS το οποίο θα ικανοποιεί καθαρά τις ανάγκες του υποσυστήματος δημοσίευσης από όπου οι πελάτες λαμβάνουν την αποδελτίωσή τους. Η εφαρμογή που παρουσιάζεται στα επόμενα κεφάλαια γεφυρώνει τα δύο αυτά υποσυστήματα.

Κεφάλαιο 3: Περιγραφή των δεδομένων της παραγωγικής βάσης και του παραγόμενου από το παραγωγικό υποσύστημα XML.

3.1 Εισαγωγή

Στο παρόν κεφάλαιο θα δοθεί μια γενική περιγραφή της λογικής δομής του παραγωγικού υποσυστήματος εστιάζοντας στα δεδομένα τα οποία διαχειρίζεται και αποθηκεύονται στο Oracle DMBS. Ο σκοπός για τον οποίο κάνουμε την περιγραφή είναι για να γίνουν καλύτερα κατανοητά τα tags τα οποία υπάρχουν στο XML αρχείο, το οποίο παράγεται από το παραγωγικό υποσύστημα. Το XML μαζί με το PDF αρχείο είναι αυτά τα οποία χρειάζονται έτσι ώστε, η εφαρμογή στην οποία αναφέρεται η παρούσα πτυχιακή, να μπορέσει να ενημερώσει το υποσύστημα δημοσίευσης με την ημερήσια αποδελτίωση.

3.2 Περιγραφή των δεδομένων της βάσης

3.2.1 Πελάτες και όροι αναζήτησης

Βασικό κομμάτι της βάσης είναι το πελατολόγιο. Ο κάθε πελάτης της εταιρείας ο οποίος λαμβάνει αποδελτίωση εντύπων αντιστοιχίζεται με έναν κωδικό. Ο κωδικός αυτός αποτελείται από μερικούς χαρακτήρες που σχετίζονται με την κατηγορία του αντικειμένου της επιχείρησης, και έναν αύξων αριθμό, προκειμένου ο κωδικός αυτός να είναι μοναδικός. Για παράδειγμα, αν υπήρχε ο πελάτης «Μεταλλουργική» που ασχολείται με την μεταλλουργία, θα είχε κωδικό το «ΜΤΛ3». Επιπλέον εισάγεται η επωνυμία της επιχείρησης του πελάτη, το ονοματεπώνυμο του υπευθύνου επικοινωνίας, τηλέφωνο, διεύθυνση, email, ημερομηνία έναρξης αποδελτίωσης και ημερομηνία λήξης της συνδρομής (εφόσον έχει οριστεί).

Πριν την έναρξη της συνδρομής ο πελάτης ορίζει το θεματολόγιό του, το οποίο βασίζεται σε κάποιες «λέξεις κλειδιά» οι οποίες εντάσσονται στις λεγόμενες «γενικές κατηγορίες». Η γενική κατηγορία δεν είναι τίποτα άλλο παρά ένα όνομα το οποίο δίνεται σε ένα σύνολο λέξεων κλειδιών, ομαδοποιώντας τις λέξεις κλειδιά που νοηματικά αναφέρονται στο ίδιο πράγμα. Οι λέξεις κλειδιά ορίζονται με την βοήθεια των regular expressions (κανονικές εκφράσεις). Αυτό είναι απαραίτητο επειδή αν ορίζαμε την λέξη κλειδί κατά απόλυτο τρόπο, έστω «εργασία», δεν θα έφερνε ως αποτέλεσμα την λέξη «εργασίας», ενώ θα την θέλαμε. Επίσης το OCR μπορεί να μην αναγνωρίσει κάποιον χαρακτήρα σωστά (επειδή δεν έχει τυπωθεί καλά, ή επειδή δεν έγινε ικανοποιητική σάρωση της σελίδας) πράγμα που θα μπορούσε να έχει σαν αποτέλεσμα την μη αξιόπιστη αναζήτηση. Οι κανονικές εκφράσεις βοηθούν κατά πολύ και προς αυτήν την κατεύθυνση. Συνοψίζοντας, μέσω των regular expressions μπορούμε να βρούμε με ακρίβεια τα σημεία του κειμένου τα οποία περιέχουν το ζητούμενο περιεχόμενο ακόμα και αν η λέξη αυτή δεν έχει αναγνωριστεί σωστά από το OCR ή ακόμα και στην περίπτωση που βρίσκεται σε διαφορετική συντακτική μορφή.

Για παράδειγμα η γενική κατηγορία, «ΕΡΓΑΤΙΚΑ», για τον πελάτη «Μεταλλουργική» περιλαμβάνει τα ακόλουθα:

- (ΕΡΓΑΣ* or Εργασ* or εργασ*) w/10 (ΣΥΜΒΑΣ* or Σ?μβ?σ* or σ?μβ?σ*)
- (ΕΡΓΑΣ* or Εργασ* or εργασ*) w/10 (ΑΤΥΧΗΜ* or ατ?χ?μ* or Ατ?χ?μ*)
- (ΕΡΓΑΣ* or Εργασ* or εργασ*) w/10 (ΩΡΑΡ* or ωράρι* or ωραρί* or Ωράρι* or Ωραρί*)
- (ΕΡΓΑΣ* or Εργασ* or εργασ*) w/10 (ΥΠΕΡΩΡ* or Υπερωρ* or υπερωρ*)
- (ΕΡΓΑΤ* or Εργ?τ* or εργ?τ*) w/10 (ΣΥΜΒΑΣ* or Σ?μβ?σ* or σ?μβ?σ*)
- (ΕΡΓΑΤ* or Εργ?τ* or εργ?τ*) w/10 (ΑΤΥΧΗΜ* or Ατ?χ?μ* or ατ?χ?μ*)

- (ΕΡΓΑΤ* or Εργ?τ* or εργ?τ*) w/10 (ΩΡΑΡΙ* or Ωράρι* or Ωραρί* or ωράρι* or ωραρί*)
- (ΕΡΓΑΤ* or Εργ?τ* or εργ?τ*) w/10 (ΥΠΕΡΩΡ* or Υπερωρ* or υπερωρ*)
- (ΕΡΓΑΤ* or Εργ?τ* or εργ?τ*) w/10 (ΔΙΑΚΡΙΣ* or διακρίσ* or διάκρισ* or Διακρίσ* or Διάκρισ*)
- (Εργαζ?μ* or εργαζ?μ* or ΕΡΓΑΖΟΜ*) w/10 (Διάκρισ* or Διακρίσ* or διάκρισ* or διακρίσ* or ΔΙΑΚΡΙΣ*)

Να σημειώσουμε ότι το αστεράκι αντικαθιστά έναν ή περισσότερους χαρακτήρες, το αγγλικό ερωτηματικό έναν χαρακτήρα ακριβώς, και το «w/10» ορίζει πόσες το πολύ λέξεις μακριά πρέπει να τοποθετείται στο κείμενο ο επόμενος όρος, δηλαδή, σε απόσταση 10 λέξεων. Έτσι, στο παράδειγμά μας, οτιδήποτε βρεθεί στο κείμενο σχετικό με εργασιακές συμβάσεις, εργασιακά ατυχήματα, εργασιακά ωράρια, εργασιακές υπερωρίες, εργατικές συμβάσεις κλπ θα αποδελτιώνεται για την γενική κατηγορία «εργατικά».

Οι γενικές κατηγορίες με την σειρά τους ομαδοποιούνται σε agreements. Μια διαφημιστική εταιρεία για παράδειγμα, η οποία με τη σειρά της έχει πελάτες άλλες εταιρείες, και παρακολουθεί την αποδελτίωση τύπου για κάθε μία από αυτές, θα έχει ως αποτέλεσμα την δημιουργία ενός agreement για τον κάθε πελάτη της. Εντός του κάθε agreement υπάρχουν οι γενικές κατηγορίες που αφορούν συγκεκριμένο πελάτης της διαφημιστικής εταιρείας και μόνο. Όταν δεν υπάρχει η ανάγκη ύπαρξης agreement για κάποια εταιρεία (όταν δηλαδή παίρνει αποδελτίωση μόνο για την ίδια) τότε δημιουργείται ένα και μοναδικό agreement με την γενική ονομασία «Άρθρα».

3.2.2 Άρθρα και Αποκόμματα

Εδώ θα κάνουμε μια μικρή παρένθεση για να ορίσουμε την διαφορά μεταξύ των όρων «άρθρο» και «απόκομμα». Άρθρο, ορίζεται ως η αυτούσια μορφή του κειμένου η οποία δημοσιοποιήθηκε στον τύπο και είναι αυτό το οποίο ονομάζουμε άρθρο εφημερίδας ή περιοδικού, ενώ απόκομμα, είναι η επεξεργασμένη μορφή του άρθρου το οποίο είναι μια ξεχωριστή οντότητα από το έντυπο. Το PDF το οποίο παράγεται από το παραγωγικό υποσύστημα

αποτελεί ένα απόκομμα και το οποίο όπως είδαμε περιέχει το άρθρο μαζί με επιπρόσθετες πληροφορίες.

Κάθε άρθρο αντιστοιχίζεται σε έναν μοναδικό κωδικό (id). Ένα άρθρο μπορεί να απευθύνεται σε πολλούς πελάτες (όχι απαραίτητα για την ίδια λέξη κλειδί).

Εναλλακτικά μπορούμε να ορίσουμε ως απόκομμα, το άρθρο, το οποίο έχει παραχθεί για συγκεκριμένο πελάτη και φυσικά, δίνονται οι επιπρόσθετες πληροφορίες σε αυτό. Από ένα άρθρο μπορούν να προκύψουν πολλά αποκόμματα αφού το άρθρο αυτό μπορεί να αφορά πολλούς πελάτες. Στην *Εικόνα 5: Δείγμα αποκόμματος αποδελτίωσης*, σελ. 11, απεικονίζεται ένα απόκομμα το οποίο περιέχει το άρθρο το οποίο ενδιαφέρει τον πελάτη, συν τις επιπρόσθετες πληροφορίες του αποκόμματος πάνω αριστερά.

Τέλος, πρέπει να σημειώσουμε ότι, με βαση το περιεχόμενο του άθρου του εντύπου το κατατάσουμε είτε στην κατηγορία «άρθρο» είτε στην κατηγορία «δημοπρασία».

3.2.3 Περιγραφή δεδομένων αρχείου XML

Πλέον μπορούμε να περιγράψουμε και να αναλύσουμε τα δεδομένα που υπάρχουν στα αρχεία XML.

Αναλύοντας τα tags, στο XML αρχείο βρίσκουμε πληροφορίες όπως

Σχετικά με το απόκομμα:

- τον μοναδικό κωδικό του κειμένου του άρθρου/δημοπρασίας (id)
- την σελίδα του εντύπου στην οποία βρίσκεται το άρθρο
- το πλήθος των σελίδων που καλύπτει το άρθρο στο έντυπο μέσο
- το μέγεθος του άρθρου σε τετραγωνικά εκατοστά
- το κείμενο OCR του άρθρου
- τον τύπο του αποκόμματος, ο οποίος διαχωρίζεται σε άρθρα και δημοπρασίες

- τον φορέα, το ποσό και την ημερομηνία λήξης της δημοπρασίας (η πληροφορία αυτή υπάρχει μόνο στην περίπτωση που το κείμενο δεν είναι άρθρο αλλά δημοπρασία)
- τον τίτλο και αρθρογράφο του άρθρου (η πληροφορία αυτή υπάρχει μόνο στην περίπτωση που το κείμενο δεν είναι δημοπρασία αλλά άρθρο)

Σχετικά με το έντυπο (στο οποίο βρίσκεται το άρθρο):

- το όνομα του εντύπου
- ο κωδικός του εντύπου
- οι ημερομηνίες έκδοσης του εντύπου
- η κυκλοφορία του εντύπου (δηλαδή πόσα αντίτυπα παρήχθησαν την ημερομηνία έκδοσής του)
- ο τύπος του εντύπου (εφημερίδα/περιοδικό/ένθετο … , οικονομική/πολιτική …)

Σχετικά με τον πελάτη:

- ο κωδικός του πελάτη
- η επωνυμία του πελάτη
- τον υπεύθυνο της εταιρείας και άλλα στοιχεία της εταιρείας
- την λέξη κλειδί την οποία αφορά το απόκομμα.
- τον κωδικό της λέξης κλειδί.
- τον κωδικό και ονομασία του Agreement στο οποίο ανήκει η λέξη κλειδί
- την ημερομηνία έναρξης και λήξης (αν έχει οριστεί) της σύμβασης του πελάτη

Επιπρόσθετα:

- την ώρα που δημιουργήθηκε το XML αρχείο.
- το όνομα του PDF αρχείου στο οποίο αναφέρεται το XML

Το XML και το PDF αρχείο έχουν το ίδιο όνομα (filename) και είναι μοναδικά. Έτσι είναι εύκολη η αναγνώριση του PDF στο οποίο αναφέρεται το XML. Φυσικά, και το DJVU το οποίο θα δημιουργηθεί από το PDF θα έχει και αυτό το ίδιο όνομα αρχείου.

Τα αποκόμματα, όπως ήδη αναφέραμε, χωρίζονται σε άρθρα και δημοπρασίες. Στην περίπτωση που το περιεχόμενο του αποκόμματος δεν είναι άρθρο αλλά είναι δημοπρασία ακολουθούνται οι ακόλουθες συμβάσεις στο αρχείο XML:

Στην σήμανση (tag) του τίτλου υπάρχει το ποσό της δημοπρασίας, μία κάθετος, και ακολουθεί η ημερομηνία λήξης της δημοπρασίας πάντα με δεδομένη μορφή (παράδειγμα <headline>3810752/20-07-2010</headline>). Επίσης αναγράφεται ο φορέας της δημοπρασίας στο αντίστοιχο tag, το agreement ονομάζεται «Δημοπρασίες» (όλοι οι πελάτες που παίρνουν δημοπρασίες έχουν ένα τέτοιο agreement) και η λέξη κλειδί αρχίζει πάντα με τον χαρακτήρα «Δ». Αξίζει να σημειώσουμε ότι ο τύπος του αποκόμματος έχει τον αριθμό 1 αν είναι άρθρο, ενώ αν είναι δημοπρασία, των αριθμό 2.

Στην περίπτωση που το απόκομμα είναι άρθρο, στην σήμανση τίτλου μπαίνει ο τίτλος του άρθρου και η σήμανση του φορέα δεν περιέχει δεδομένα.

Αρχείο XML άρθρου και δημοπρασίας με την αναλυτική περιγραφή του κάθε tag μπορείτε να βρείτε στο παράρτημα Α (*Δείγμα αρχείου XML – Άρθρο (σελ. 108), Δείγμα αρχείου XML - Δημοπρασία (σελ. 109), Ανάλυση σημάνσεων (tags) (σελ. 110*).

3.3 Σύνοψη

Είδαμε σε γενικές γραμμές το είδος των δεδομένων που υπάρχουν στην βάση στην οποία στηρίζεται το υποσύστημα παραγωγής. Το χρήσιμο (για την βάση του υποσυστήματος δημοσίευσης) υποσύνολο δεδομένων τα οποία αφορούν το κάθε απόκομμα, εξάγεται αυτόματα σε αρχείο XML ταυτόχρονα με την δημιουργία του. Έγινε ανάλυση αυτών των δεδομένων όπως και των ιδιαιτεροτήτων τους.

Κεφάλαιο 4: Σχεδίαση της βάσης του υποσυστήματος δημοσίευσης.

4.1 Πρόλογος

Η εφαρμογή η οποία παρουσιάζεται στην παρούσα πτυχιακή εργασία είναι αυτή η οποία θα διασυνδέει το παραγωγικό υποσύστημα με το υποσύστημα δημοσίευσης σε εταιρεία αποδελτίωσης εντύπων. Το δεύτερο τμήμα της παρούσας πτυχιακής εργασίας έχει να κάνει με τη σχεδίαση και τη δημιουργία της βάσης που θα υποστηρίζει το υποσύστημα δημοσίευσης. Θα ακολουθήσει περιγραφή της λογικής και των προδιαγραφών της βάσης που θα υλοποιηθεί.

4.2 Ανάλυση απαιτήσεων και εννοιολογικός σχεδιασμός βάσης.

Το πρώτο βήμα στην σχεδίαση μιας βάσης είναι η ανάλυση των απαιτήσεων, δηλαδή η περιγραφή των δεδομένων και των συνθηκών που την συνοδεύουν. Ακολουθεί ο *Πίνακας 1: Πίνακας εννοιών* με τα απαραίτητα για το υποσύστημα δημοσίευσης στοιχεία τα οποία είναι διαθέσιμα μέσω του αρχείου XML. Στη συνέχεια δίνεται η περιγραφή των συνθηκών που συνοδεύουν αυτά τα δεδομένα και τέλος, το διάγραμμα ER το οποίο προκύπτει από αυτά στοιχεία.

Πίνακας 1: *Πίνακας εννοιών*

Όνομα Πεδίου	Επεξήγηση
Κωδ_εταιρείας	Κωδικός εταιρείας: Ο τομέας απασχόλησης της εταιρείας σε συντομογραφία ο οποίος ακολουθείται από έναν αύξον αριθμό. πχ. ΜΤΛ3 Τα επιμέρους στοιχεία δεν μας ενδιαφέρουν ως ξεχωριστές οντότητες. Μας ενδιαφέρει ο κωδικός στο σύνολό του.
Όνομα_εταιρείας	Όνομα εταιρείας πελάτη πχ. Μεταλλουργική Α.Ε.

Ημερ_λήξης	Ημερομηνία λήξης συνδρομής του πελάτη.
Username	Το συνθηματικό όνομα εισόδου χρήστη για τον λογαριασμό αποδελτίωσης του πελάτη. Πρέπει να είναι μοναδικό.
Password	Ο συνθηματικός κωδικός εισόδου χρήστη για τον λογαριασμό αποδελτίωσης του πελάτη.
Ονομ/νυμο_χρήστη	Όνομα και επώνυμο του χρήστη (πχ Γιάννης Ταρσιάς). Η κάθε εταιρεία μπορεί να έχει έναν ή περισσότερους χρήστες οι οποίοι παρακολουθούν την αποδελτίωση. Ο κάθε χρήστης έχει ένα συνθηματικό όνομα εισόδου και ένα συνθηματικό κωδικό εισόδου (τα οποία αναφέραμε πιο πάνω).
Email	Το email του χρήστη.
Πόλη	Πόλη διαμονής του χρήστη.
Όνομα_αρχείου	Το όνομα αρχείου αποκόμματος χωρίς την κατάληξη (πχ. 123456). Το όνομα αρχείου είναι η αναφορά στο απόκομμα το οποίο βρίσκεται κάτω από συγκεκριμένη διεύθυνση (πχ C:\root\PDF\123456.pdf). Να σημειώσουμε πως χρειαζόμαστε αυτό το πεδίο γιατί τα αποκόμματα αυτά καθ' αυτά (PDF και DJVU αρχεία) δεν θα εισάγονται στην βάση (BLOB type) αλλά θα υπάρχουν αναφορές σε αυτά. Το κάθε όνομα αρχείου είναι μοναδικό. Το κάθε απόκομμα ανήκει σε έναν και μόνο πελάτη.
Ημερ_δημιουργίας	Ημερομηνία δημιουργίας αποκόμματος
Ώρα_δημιουργίας	Ώρα δημιουργίας αποκόμματος
Σελίδα	Η σελίδα του εντύπου στην οποία βρέθηκε το άρθρο ή η δημοπρασία.
Αρ_σελίδων	Το πόσες σελίδες είναι το άρθρο ή η δημοπρασία.
Μέγεθος_σελίδας	Το μέγεθος (A3, A4) του άρθρου ή της δημοπρασίας.
Τίτλος	Ο τίτλος του άρθρου στο απόκομμα.

Αρθρογράφος	Ο συντάκτης του άρθρου.
Εμβαδό	Το μέγεθος του άρθρου ή της δημοπρασίας (μόνο το κείμενο/στήλη που έχουμε πάρει από το έντυπο, σε cm^2).
Κείμενο	Το OCR κείμενο.
Κωδικός_Κειμένου	Ο μοναδικός κωδικός του κάθε κειμένου. Ο κωδικός αυτός θα είναι ίδιος μεταξύ των δύο υποσυστημάτων για να μπορεί να γίνεται η ταυτοποίηση των κειμένων. Αν δηλαδή το κείμενο με id 301 έχει εισαχθεί στο υποσύστημα δημοσίευσης, να μην εισαχθεί ξανά.
Όνομα_φορέα	*Το όνομα του φορέα (μόνο για δημοπρασίες)*
Ποσό_δημοπρ	Το ποσό της δημοπρασίας (μόνο για δημοπρασίες).
Ημ_λήξης_δημοπρ	Η ημερομηνία λήξης της δημοπρασίας (μόνο για δημοπρασίες).
ΛΚ	Ο κάθε πελάτης έχει μία ή και περισσότερες γενικές κατηγορίες. Οι γενικές κατηγορίες του πελάτη για λόγους απλότητας θα αναφέρονται από δω και πέρα ως λέξεις κλειδιά, επειδή ως λέξεις κλειδιά παρουσιάζονται και στον πελάτη.
Κωδικός_ΛΚ	Ο μοναδικός κωδικός για την κάθε λέξη κλειδί. Ο κωδικός αυτός θα είναι ίδιος μεταξύ των δύο υποσυστημάτων έτσι ώστε να είναι δυνατή η ταυτοποίηση των λέξεων κλειδιών. Αν δηλαδή μετονομαστεί κάποια λέξη κλειδί στο παραγωγικό υποσύστημα, με κωδικό 100, η λέξη κλειδί με κωδικό 100 στο υποσύστημα δημοσίευσης θα υποστεί επίσης μετονομασία.
Όνομα_εντύπου	Το όνομα εντύπου (πχ «Η καθημερινή»).
Κωδικός_εντύπου	Ο μοναδικός κωδικός για το κάθε έντυπο. Ο κωδικός αυτός θα είναι ίδιος μεταξύ των δύο υποσυστημάτων έτσι ώστε να είναι δυνατή η ταυτοποίηση των εντύπων. Αν δηλαδή μετονομαστεί κάποιο έντυπο

	στο υποσύστημα παραγωγής, να υποστεί μετονομασία και στο υποσύστημα δημοσίευσης.
Τύπος_εντύπου	πχ. «Εφημερίδα@Κύρια@Πολιτική@ Ημερήσια».
Ημερ_έκδοσης	Ημερομηνία έκδοσης του εντύπου.
Αρ_αντιτύπων	Η κυκλοφορία του εντύπου.

Κατά κανόνα οι πελάτες εταιρείας αποδελτίωσης είναι άλλες εταιρείες. Έτσι αναφερόμενοι στο όρο «εταιρεία» θα εννοούμε τους πελάτες της εταιρείας αποδελτίωσης εντύπων.

Οι συνθήκες που συνοδεύουν τα δεδομένα του *Πίνακας 1: Πίνακας εννοιών* είναι:

- Η κάθε εταιρεία η οποία παίρνει αποδελτίωση θα έχει απαραίτητα τουλάχιστον έναν χρήστη ο οποίος θα έχει πρόσβαση στα αποκόμματα μέσω του ιστότοπου της εταιρείας αποδελτίωσης. Τα στοιχεία σύνδεσης του χρήστη (username, password) στον ιστότοπο δεν υπάρχουν στο παραγωγικό υποσύστημα αλλά υπάρχουν στο υποσύστημα δημοσίευσης και γι' αυτό δεν υπάρχουν στο XML που παράγει το παραγωγικό υποσύστημα.
- Ο κάθε χρήστης θα πρέπει να εργάζεται σε μόνο μία εταιρεία.
- Η εταιρεία λαμβάνει αποκόμματα μέχρι να επέλθει η ημερομηνία λήξης της συνδρομής της.

  Αυτή η πληροφορία υπάρχει στο παραγωγικό υποσύστημα και θα μπορούσε να εμπεριέχεται στα XML αρχεία αλλά δεν συμπεριλαμβάνεται επειδή δεν μπορεί να αξιοποιηθεί από το υποσύστημα δημοσίευσης. Συγκεκριμένα, στο παραγωγικό υποσύστημα υπάρχει συγκεκριμένη ημερομηνία λήξης για το κάθε agreement. Το κάθε agreement έχει την δική του ημερομηνία λήξης. Αν δηλαδή έχει λήξει ένα agreement, δεν σημαίνει ότι ο πελάτης έχει σταματήσει την αποδελτίωση στο σύνολό της. Είναι πολύ πιθανό να υπάρχουν πολλά ακόμα ενεργά agreement. Στο υποσύστημα δημοσίευσης όμως δεν θα υπάρχει η έννοια του agreement. Απλά ένας πελάτης θεωρείται ανενεργός ή ενεργός. Αν υπάρχει έστω και ένα ενεργό agreement για κάποιον πελάτη τότε ο

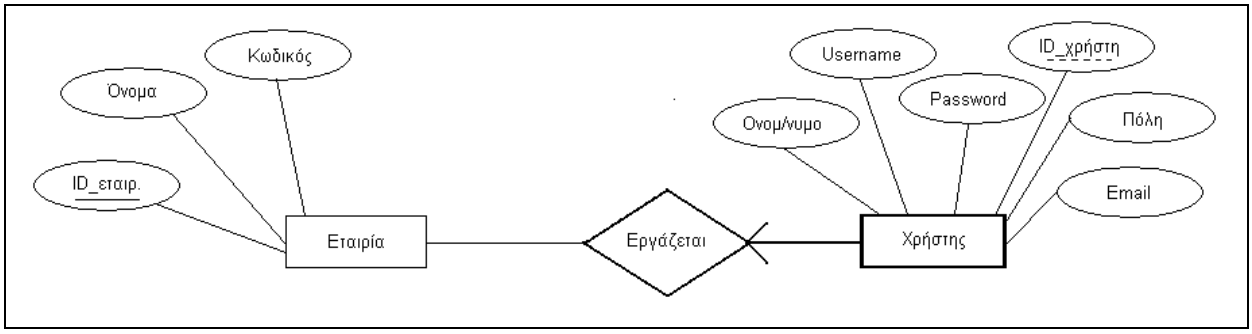
πελάτης αυτός για την βάση του υποσυστήματος δημοσίευσης θα θεωρείται ενεργός. Αντίθετα, αν όλα τα agreement του είναι ανενεργά, θα θεωρείται ανενεργός. Εδώ βλέπουμε και την ανάγκη συγχρονισμού της βάσης του παραγωγικού υποσυστήματος με την βάση του υποσυστήματος δημοσίευσης, η οποία θα ενημερώνεται από την πρώτη για το αν οι καταχωρημένοι πελάτες είναι ενεργοί ή όχι.

- Το κάθε απόκομμα ανήκει σε μόνον ένα πελάτη.
- Μία στήλη/άρθρο/δημοπρασία εφημερίδας ή περιοδικού μπορεί να κοπεί για πολλούς πελάτες με βάση το ποιες λέξεις κλειδιά, και ποιών πελατών εντοπίστηκαν στο κείμενο.
- Κατά την έναρξη της συνδρομής του ο πελάτης δεν θα έχει αποκόμματα. Παρ' αυτά, θα πρέπει αναγκαστικά να έχουν καταχωρηθεί στο υποσύστημα δημοσίευσης τα στοιχεία τουλάχιστον ενός χρήστη και να έχουν καταχωρηθεί οι λέξεις κλειδιά για τις οποίες ενδιαφέρεται.
- Η κάθε λέξη κλειδί μπορεί να αφορά έναν ή περισσότερους πελάτες.
- Το κάθε απόκομμα αναφέρεται απαραίτητα σε συγκεκριμένη λέξη κλειδί και μόνο σε μία λέξη κλειδί. Στην περίπτωση που στο απόκομμα υπάρχουν δύο ή περισσότερες αναφορές σε διαφορετικές λέξεις κλειδιά το απόκομμα θα κόβεται από το υποσύστημα παραγωγής μόνο για μία λέξη κλειδί με βάση την σειρά προτεραιότητας των λέξεων κλειδιών που έχει δώσει ο πελάτης.
- Το απόκομμα θα είναι είτε άρθρο είτε δημοπρασία.
- Το απόκομμα προέρχεται από συγκεκριμένο έντυπο, συγκεκριμένης ημερομηνίας έκδοσης.
- Το κάθε έντυπο έχει εκδοθεί τουλάχιστον μία φορά, σε συγκεκριμένη μέρα και έχει τυπωθεί σε συγκεκριμένο αριθμό αντιτύπων (κυκλοφορία εντύπου).

4.3 Διάγραμμα ER

Έχοντας τα δεδομένα και τις συνθήκες μπορούμε να χτίσουμε το μοντέλο ER.

- Εύκολα διαχωρίζει κανείς την οντότητα της **εταιρείας**, που είναι ο πελάτης. Τα γνωρίσματα της εταιρείας είναι η *ονομασία* της (πχ Μεταλλουργική Α.Ε) και ο *κωδικός* της (πχ ΜΤΛ3).
- Στην κάθε εταιρεία εργάζεται τουλάχιστον ένας **χρήστης**, ο οποίος χειρίζεται την αποδελτίωση της εταιρείας. Ο κάθε χρήστης έχει τα ακόλουθα γνωρίσματα: *ονοματεπώνυμο, username, password, πόλη, email.* Η ύπαρξη του χρήστη εξαρτάται αποκλειστικά από την ύπαρξη της εταιρείας και άρα ο χρήστης θα είναι ασθενής οντότητα.



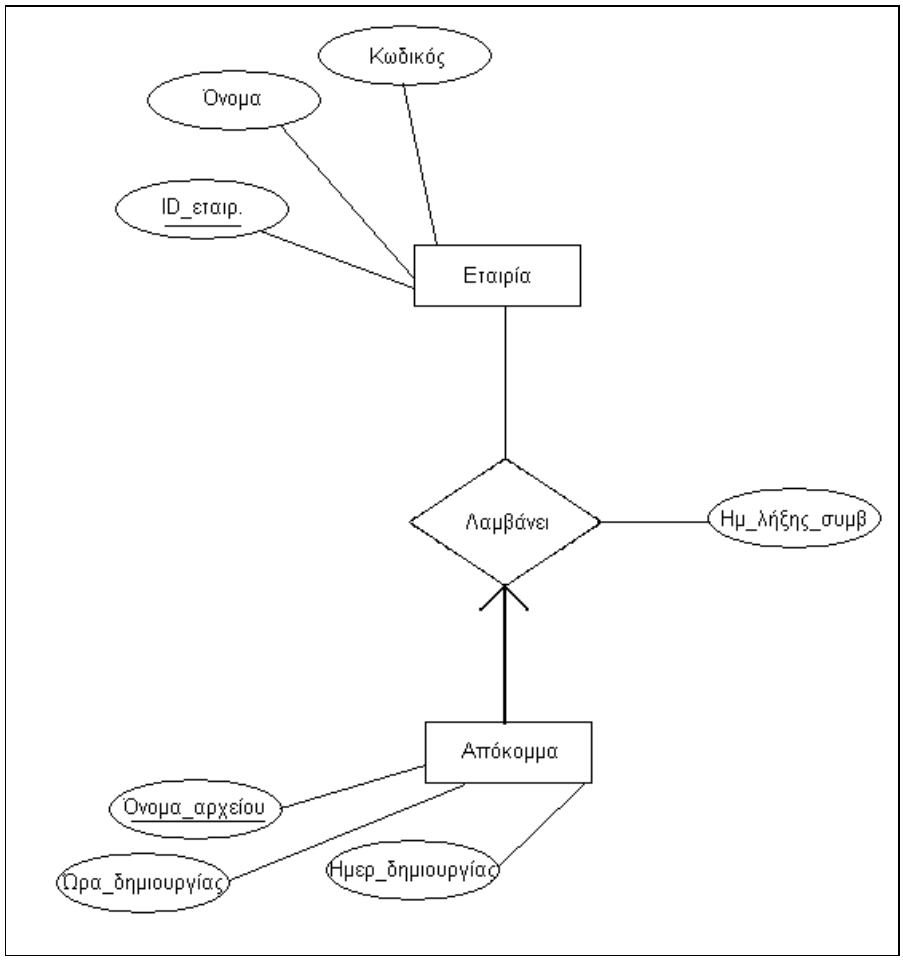
Εικόνα 8: *ER - Σχέση «Εργάζεται»*

Στο διάγραμμα *ER - Σχέση «Εργάζεται»*, προσθέσαμε και το *ID_εταιρ* το οποίο θα είναι ένα αριθμητικό κύριο κλειδί για την Εταιρεία, και το *ID_χρήστη* το οποίο θα είναι υποψήφιο κλειδί για την ασθενή οντότητα.

- Η εταιρεία λαμβάνει **αποκόμματα** από την εταιρεία αποδελτίωσης μέχρι να επέλθει *η ημερομηνία λήξης του συμβολαίου*. Το απόκομμα χαρακτηρίζεται από το *όνομα του αρχείου του,* το οποίο είναι μοναδικό μεταξύ των αρχείων, και την *ημερομηνία και ώρα δημιουργίας του*.

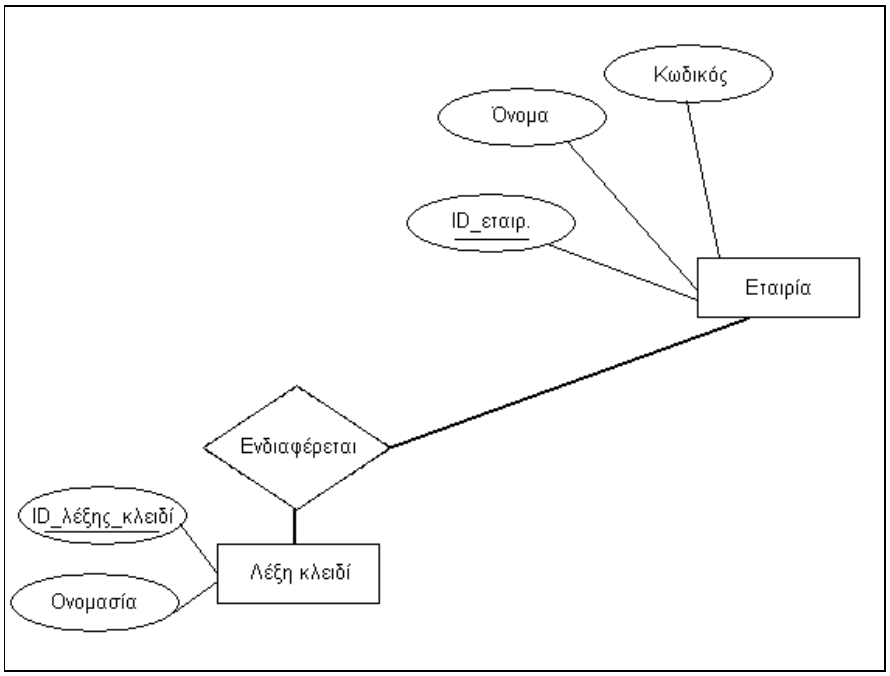
Δεν είναι απαραίτητο με την έναρξη της συνδρομής η εταιρεία να έχει απευθείας διαθέσιμα αποκόμματα.

Το απόκομμα ανήκει υποχρεωτικά σε έναν και μόνο πελάτη.



Εικόνα 9: *ER - Σχέση «Λαμβάνει»*

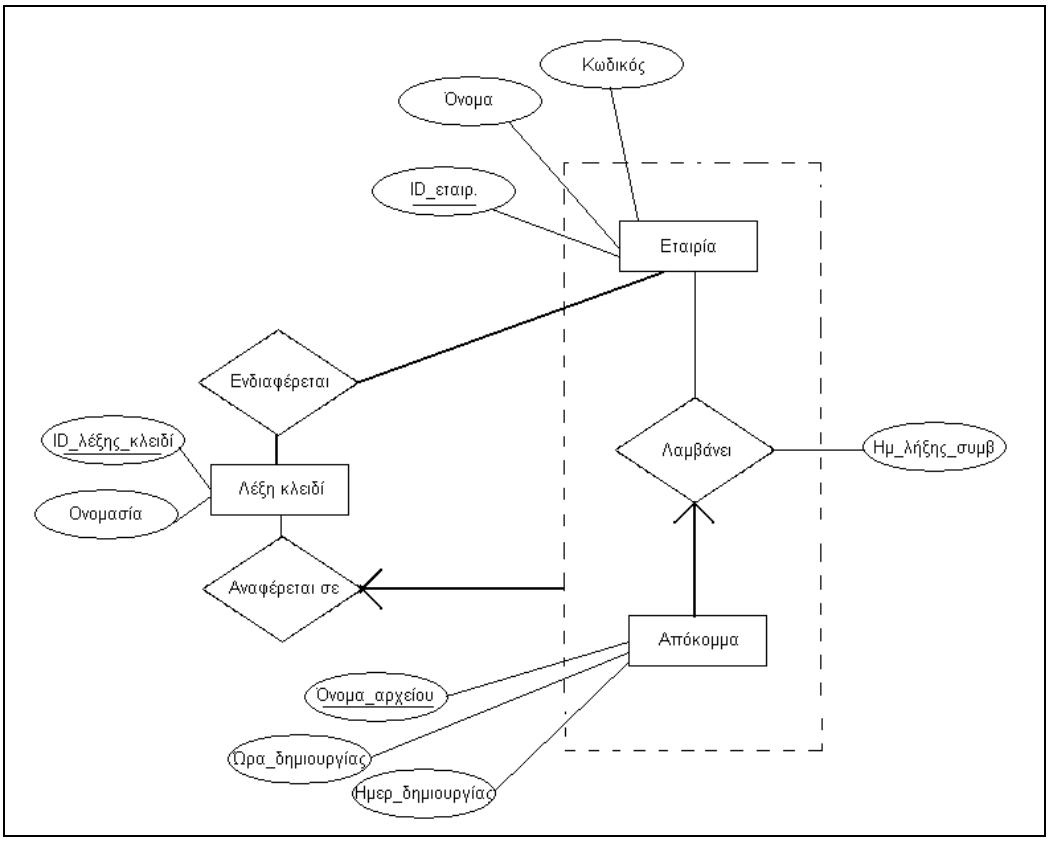
- Η εταιρεία ενδιαφέρεται για μία ή περισσότερες **λέξεις κλειδιά**. Για την ίδια λέξη κλειδί μπορούν να ενδιαφέρονται μία ή περισσότερες εταιρείες. Η λέξη κλειδί χαρακτηρίζεται από την *ονομασία* της και από τον μοναδικό *κωδικό* της, πχ ονομασία: ΠΕΤΡΕΛΑΙΟ, κωδικός: 1236.



Εικόνα 10: *ER - Σχέση «Ενδιαφέρεται»*

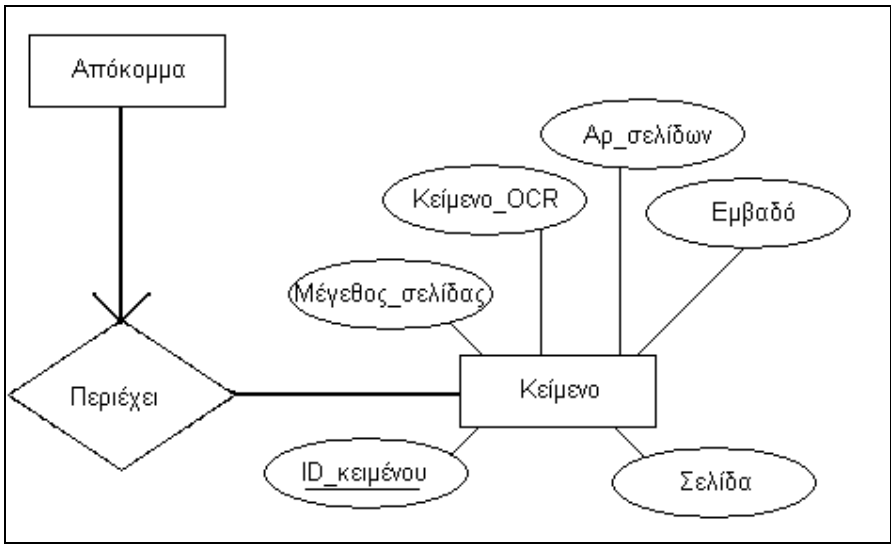
- **Το *απόκομμα συγκεκριμένης εταιρείας*** αναφέρεται σε **λέξη κλειδί** για την οποία μπορεί να ενδιαφέρονται περισσότεροι του ενός πελάτες.

Έχοντας την σχέση «Λαμβάνει» και «Αναφέρεται σε», ξέρουμε για την κάθε εταιρεία ποια είναι τα αποκόμματά της και άρα το ποιες λέξεις κλειδιά παρακολουθεί. Δείχνει δηλαδή, ότι η σχέση «Ενδιαφέρεται» πλεονάζει. Στην πραγματικότητα όμως χρειαζόμαστε και τις δύο σχέσεις γιατί θέλουμε να ξέρουμε ποιες είναι οι λέξεις κλειδιά για τις οποίες ενδιαφέρεται ο πελάτης ανεξάρτητα με το αν έχει ή όχι αποκόμματα για αυτές.



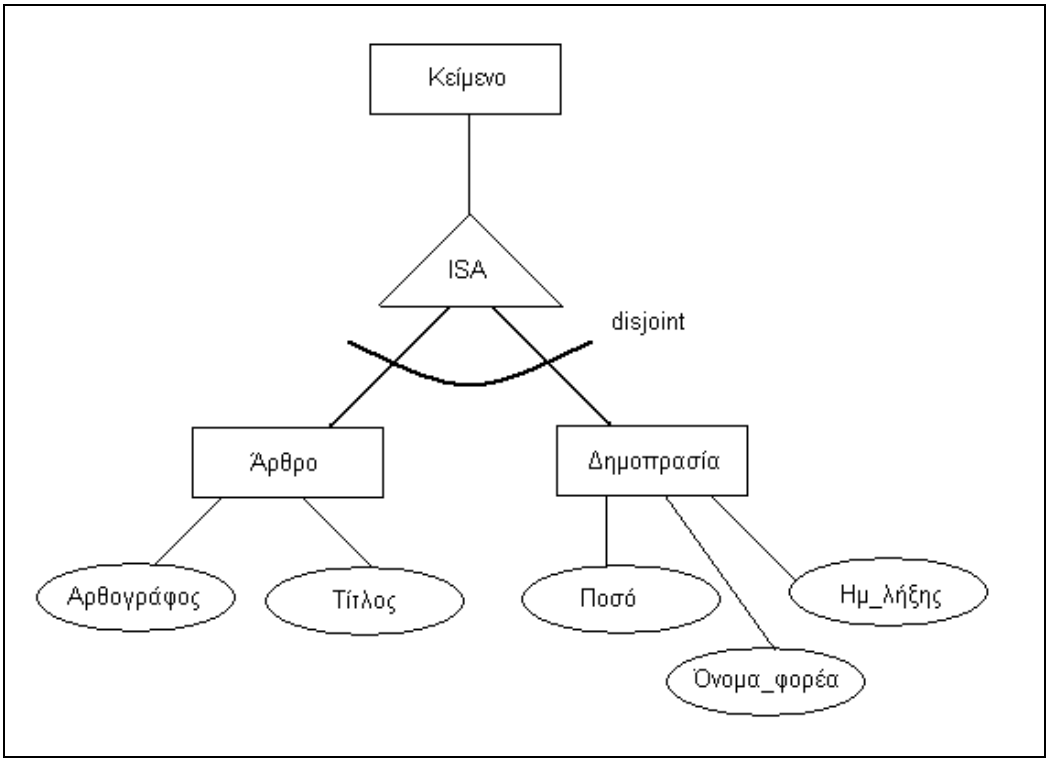
Εικόνα 11: *ER - Σχέση «Αναφέρεται σε»*

- Το **απόκομμα** περιέχει απαραίτητα συγκεκριμένο **κείμενο**, το οποίο έχει εξαχθεί από κάποιο έντυπο μέσο, και είναι η πληροφορία για την οποία ενδιαφέρεται ένας ή περισσότεροι πελάτες. Το κείμενο χαρακτηρίζεται από το την *σελίδα* στην οποία βρέθηκε, το *εμβαδό* του (δλδ την έκταση σε cm^2 την οποία καταλαμβάνει πάνω στο χαρτί), τον *αριθμό των σελίδων* που καταλαμβάνει (πχ μπορεί να συνεχίζεται και σε δεύτερη και σε τρίτη σελίδα), το *κείμενο* αυτό καθ' αυτό (OCR text) και το *μέγεθός* του σε διαστάσεις A3/A4.



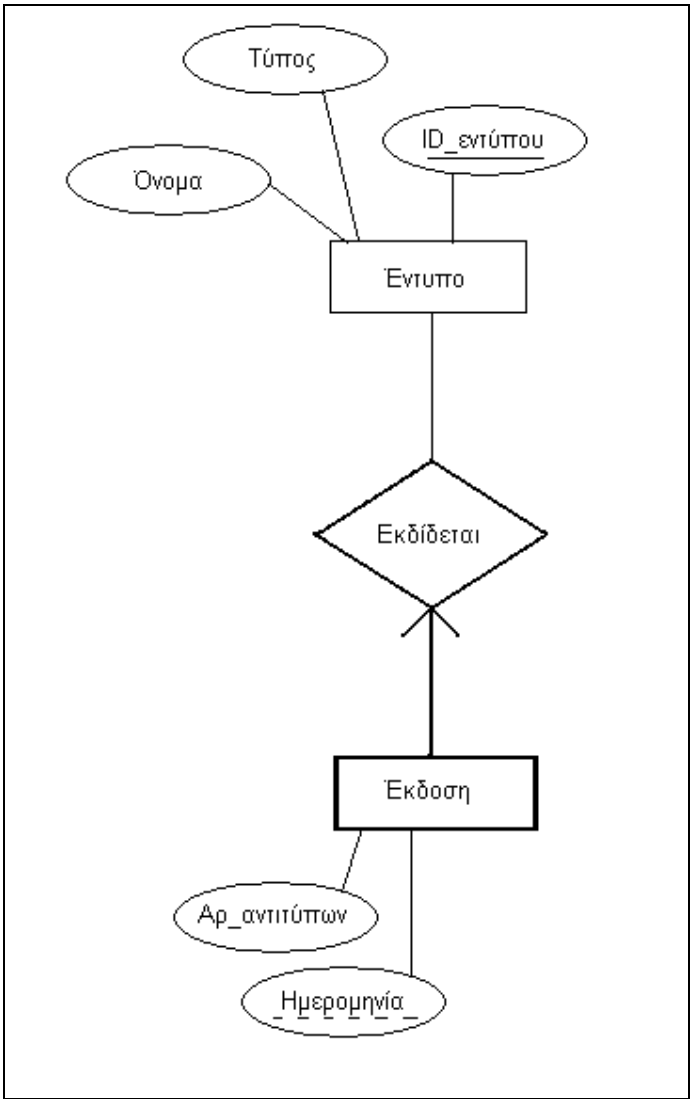
Εικόνα 12: *ER - Σχέση «Περιέχει».*

- Το κείμενο θα είναι είτε **άρθρο** είτε **δημοπρασία**. Το άρθρο χαρακτηρίζεται από τον *τίτλο* και τον *αρθογράφο* του, ενώ η δημοπρασία από το *ποσό*, τον *φορέα* και την *ημερομηνία* λήξης της.



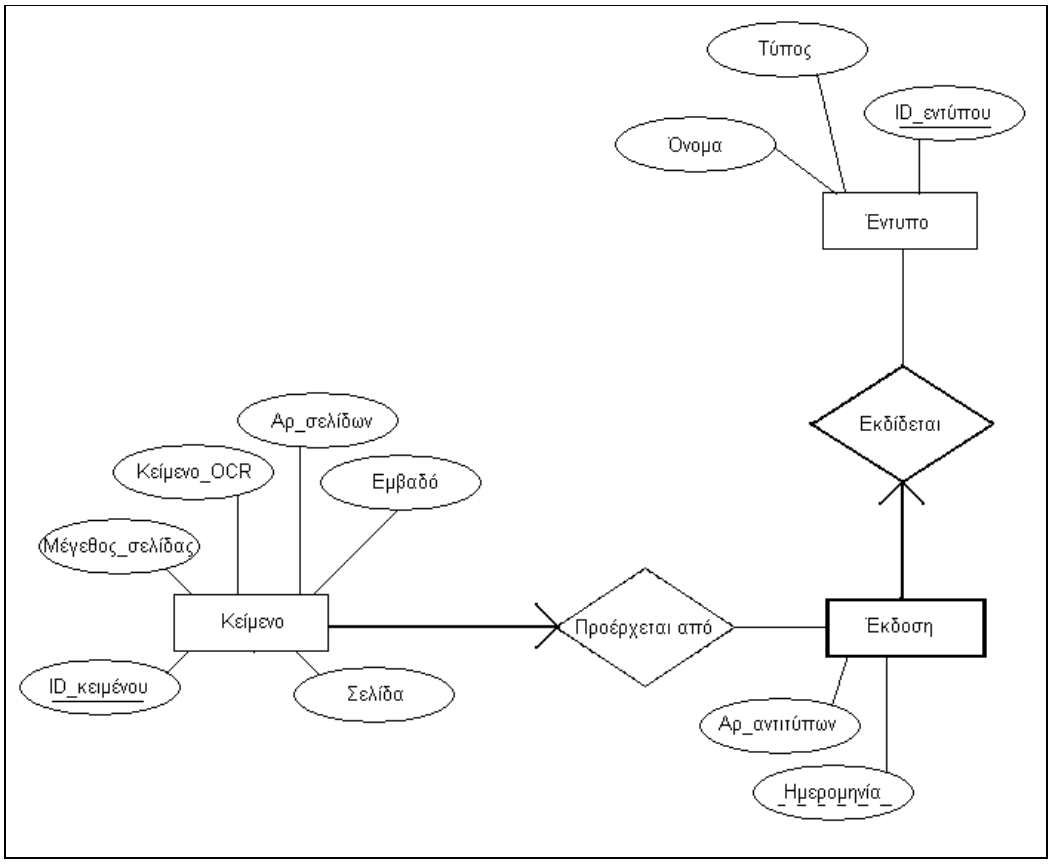
Εικόνα 13: *ER - Διαχωρισμός άρθρου/δημοπρασίας.*

- Το κάθε **έντυπο** εκδίδεται περιοδικά (πχ εβδομαδιαία, καθημερινά, μηνιαία κλπ). Η κάθε **έκδοση** του εντύπου χαρακτηρίζεται από την *ημερομηνία έκδοσης* και από τον *αριθμό των αντιτύπων*. Το έντυπο χαρακτηρίζεται από το *όνομά* του (πχ Καθημερινή) και τον *τύπο* του (πχ Οικονομική). Η έκδοση ενός εντύπου εξαρτάται πλήρως από το έντυπο (δλδ δεν μπορεί να σταθεί από μόνο του ως οντότητα) και γι' αυτό αποτελεί ασθενή οντότητα.



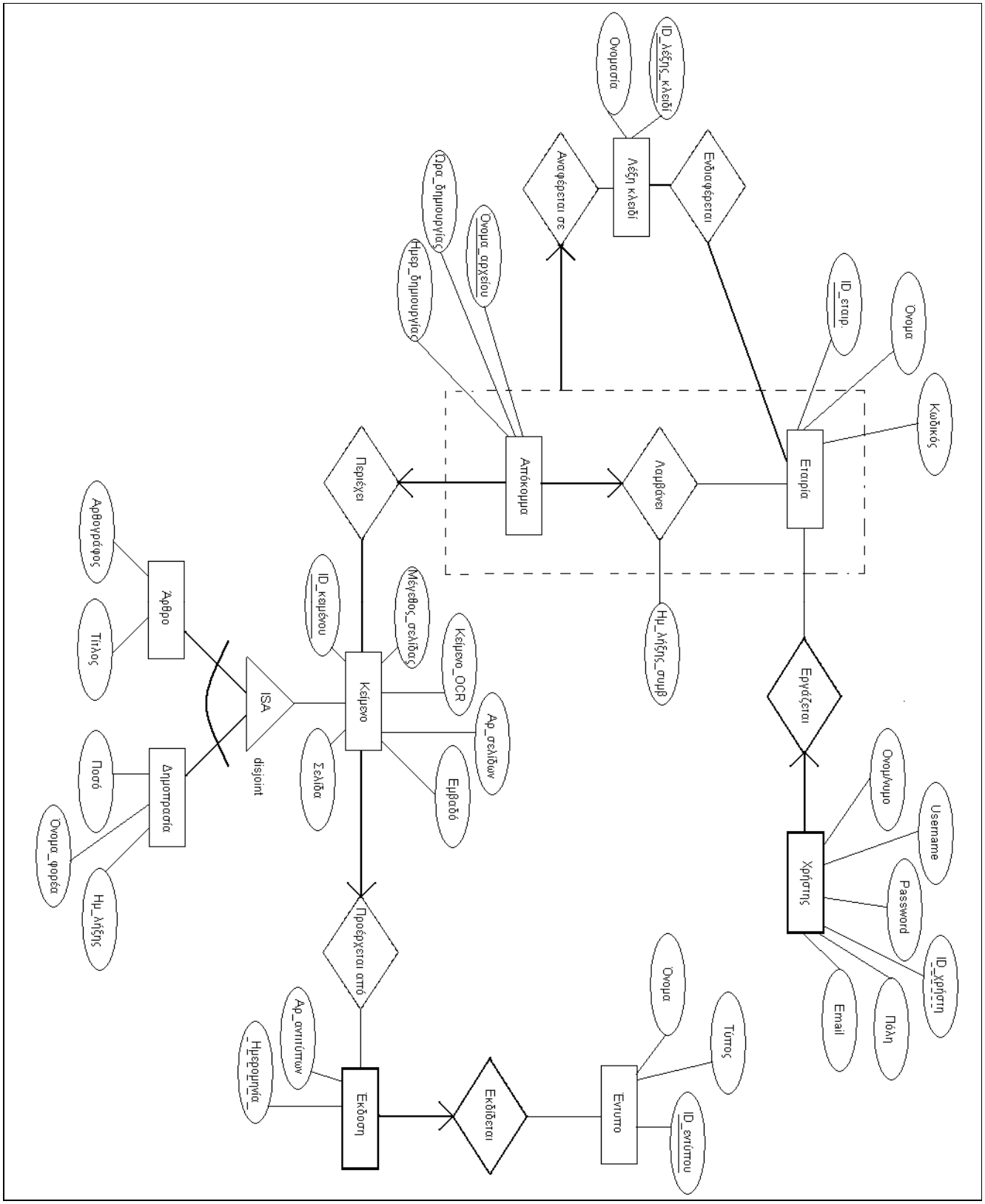
Εικόνα 14: *ER - Σχέση «Εκδίδεται»*

- Το κείμενο προέρχεται υποχρεωτικά από την συγκεκριμένη έκδοση του εντύπου.



Εικόνα 15: *ER - Σχέσεις «Προέρχεται από» και «Εκδίδεται»*

Τέλος, ενώνοντας τα κομμάτια τα οποία μόλις περιγράψαμε προκύπτει το ακόλουθο ER:



Εικόνα 16: *ER - Διάγραμμα ER*

4.4 Λογικός Σχεδιασμός βάσης

Από το διάγραμμα ER δημιουργούνται οι ακόλουθες σχέσεις:

- **Εταιρεία** (ID_εταιρείας, Κωδικός, Όνομα, ημερ_λήξης_σύμβασης)
- **Χρήστης** (ID εταιρείας, ID_χρήστη, Ονομ/νυμο, Username, Password, Πόλη, Email)
- **Λέξη_Κλειδί** (ID_λέξης_κλειδί, Ονομασία)
- **Ενδιαφέρεται** (ID_λέξης_κλειδί, ID_εταιρείας)
- **Έντυπο** (ID_εντύπου, Τύπος, Όνομα)
- **Έκδοση** (ID_έκδοσης, ID_εντύπου, Ημερομηνία, αρ_αντιτύπων)
- **Κείμενο** (ID_κειμένου, ID_έκδοσης, Κείμενο_OCR, Μέγεθος_σελίδας, Αρ_σελίδων, Εμβαδό, Σελίδα)
- **Άρθρο** (ID_κειμένου, Τίτλος, Αρθογράφος)
- **Δημοπρασία** (ID_κειμένου, Ποσό, Όνομα_φορέα, Ημ_λήξης)
- **Απόκομμα** (Όνομα_αρχείου, ID_εταιρείας, ID_Κειμένου, ID_λέξης_κλειδί, Ώρα_δημιουργίας, Ημερ_δημιουργίας).

4.5 Φυσικός Σχεδιασμός βάσης

Έχοντας τον λογικό σχεδιασμό, το επόμενο βήμα είναι ο φυσικός σχεδιασμός της βάσης. Στην παρούσα πτυχιακή χρησιμοποιήσαμε MS SQL 2008 και το Microsoft Management Studio για την διασύνδεση μας με το DBMS. Αναλυτικές οδηγίες εγκατάστασης του DBMS όπως και της δημιουργίας της βάσης μπορείτε να δείτε στο Παράρτημα Γ, Εγκατάσταση MS SQL Server (σελ. 119), Δημιουργία βάσης δεδομένων με χρήση SQL Server Management Studio.σελ. 142).

4.5.1 Πίνακες

Η αντιστοιχία των πινάκων στα διαγράμματα με αυτή του λογικού σχεδιασμού είναι η ακόλουθη:

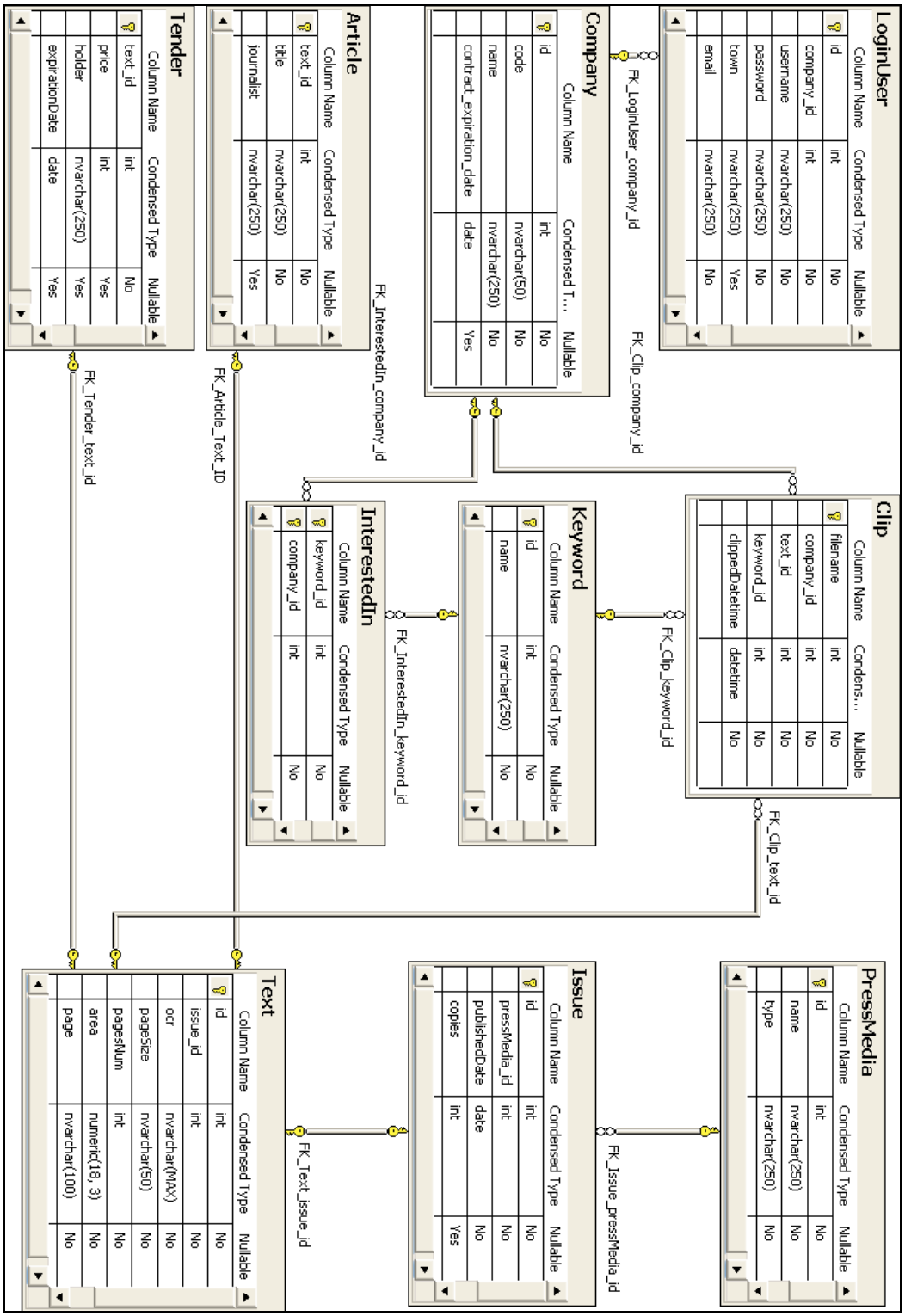
Πίνακας 2: *Αντιστοιχία ονομάτων πινάκων μεταξύ ER και Database*

Έκδοση	**Issue**
Έντυπο	**PressMedia**
Δημοπρασία	**Tender**
Άρθρο	**Article**
Κείμενο	**Text**
Εταιρεία	**Company**
Χρήστης	**LoginUser**
Ενδιαφέρεται	**InterestedIn**
Λέξη κλειδί	**Keyword**
Απόκομμα	**Clip**

Στο διάγραμμα στην *Εικόνα 17: Διάγραμμα πινάκων της βάσης του υποσυστήματος δημοσίευσης* φαίνονται τα ξένα κλειδιά, οι αναφορές σε ξένα κλειδιά, ο τύπος του κάθε πεδίου, και το αν είναι nullable ή όχι. Ο τύπος nvarchar χρησιμοποιείται για Unicode, αλφαριθμητικές μεταβλητές μεταβλητού μήκους.

Οι DML εντολές για την δημιουργία των πινάκων υπάρχουν στο παράρτημα Δ, Δημιουργία πινάκων με SQL εντολές., σελ. 147.

Αξίζει να σημειώσουμε πως στους πίνακες υπάρχουν δύο ειδών id. Τα περισσότερα id είναι τύπου auto increment, δηλαδή αποτελούν έναν αύξων αριθμό. Όμως τα id των πινάκων Keyword, PressMedia και Text δεν είναι auto increment αλλά παίρνουν την τιμή id (auto increment) που έχουν οι αντίστοιχες εγγραφές της βάσης του παραγωγικού υποσυστήματος, η οποία τιμή είναι μοναδική. Είναι δηλαδή “ξένο κλειδί” του αντίστοιχου πεδίου της βάσης του παραγωγικού υποσυστήματος. Ο λόγος που το κάνουμε αυτό είναι γιατί για αυτούς τους τρείς πίνακες υπάρχει η ανάγκη της αντιστοίχησης των εγγραφών για να μπορούμε να προβούμε σε προσθήκες ή αλλαγές στις εγγραφές. Αν για παράδειγμα ένα έντυπο από την βάση του παραγωγικού υποσυστήματος με id 456 και όνομα «Καθημερινή» αλλάξει όνομα σε «Η Καθημερινή» (και φυσικά συνεχίζει να έχει το ίδιο id), τότε χωρίς την πληροφορία του id της βάσης του παραγωγικού υποσυστήματος η εφαρμογή δεν θα μπορεί να ξέρει ότι η αλλαγή αυτή αφορά υπάρχον έντυπο, αφού δεν θα μπορεί να κάνει την ταυτοποίηση με βάση το όνομα. Αυτό που θα θεωρήσει είναι ότι το «Η Καθημερινή» είναι ένα νέο έντυπο το οποίο θα πρέπει να το εισάγει στον πίνακα μαζί με τα υπόλοιπα έντυπα, πράγμα που είναι λάθος. Όμοια, για να μπορούμε να ξέρουμε ποιες λέξεις κλειδιά προστίθενται στον πελάτη, ποιες λέξεις κλειδιά αφαιρούνται και ποιες λέξεις κλειδιά έχουν μετονομαστεί/αλλαχθεί, θα πρέπει να κρατάμε και το auto increment id της κάθε λέξεις κλειδί της βάσης του παραγωγικού υποσυστήματος στην βάση του υποσυστήματος δημοσίευσης έτσι ώστε να μπορούμε να αντιστοιχίσουμε τις λέξεις κλειδιά.



Εικόνα 17: *Διάγραμμα πινάκων της βάσης του υποσυστήματος δημοσίευσης*

4.5.2 Stored procedures

Με βάση τις προδιαγραφές που έχουμε συζητήσει υπάρχουν μερικά σημεία τα οποία δεν έχουν υλοποιηθεί και είναι τα ακόλουθα:

1. Δεν υπάρχει περιορισμός ο οποίος να μην επιτρέπει την ύπαρξη id στον πίνακα Text, για τον οποίο θα υπάρχει αναφορά και στον πίνακα Article και στον πίνακα Tender. Δεν μπορεί δηλαδή ένα απόκομμα να είναι και άρθρο και δημοπρασία, και η ως τώρα σχεδίαση δεν υποστηρίζει αυτό τον περιορισμό.
2. Θα μπορούσε κάποιος από λάθος ή από μη καλή γνώση της λογικής των πινάκων να εισάγει εγγραφή στον πίνακα Company και να μην εισάγει τον αρχικό χρήστη (LoginUser) και τις λέξεις κλειδιά για τις οποίες ενδιαφέρεται ο πελάτης (Keywords, InterestedId). Το ίδιο ισχύει για όλες τις υποχρεωτικές συσχετίσεις.
3. Θα πρέπει να εξασφαλιστεί πως οι λέξεις κλειδιά για τις οποίες έχει πάρει αποκόμματα ο πελάτης (όπως και για αυτές που δεν έχει πιθανώς πάρει αποκόμματα) είναι καταχωρημένες στον πίνακα InterestedIn.

Ο πιο εύκολος, εύχρηστος και αποδοτικός (από άποψη performance) τρόπος για να αποφευχθούν όλες αυτές οι καταστάσεις είναι να απαγορεύσουμε την απευθείας εκτέλεση Insert, Update, Delete εντολών στους πίνακες και να δημιουργηθούν τα κατάλληλα stored procedures τα οποία θα διαχειρίζονται τις εντολές που πρέπει να πραγματοποιηθούν. Για παράδειγμα, αν υπάρχει ένα stored procedure το οποίο θα κάνει Insert τα άρθρα ενημερώνοντας κατάλληλα και τους υπόλοιπους πίνακες, θα έχει ως συνέπεια την αποφυγή οποιουδήποτε λάθους που θα μπορούσε να συμβεί κατά την εισαγωγή άρθρου από λάθος χειρισμό. Ουσιαστικά εισάγουμε ένα ενδιάμεσο επίπεδο μεταξύ των πινάκων και της εφαρμογής μέσω του οποίου γίνεται η διαχείριση των δεδομένων στους πίνακες. Έτσι ο προγραμματιστής δεν θα μπορεί να κάνει Insert σε κάποιον πίνακα αλλά θα είναι αναγκασμένος να χρησιμοποιήσει στο stored procedure το οποίο θα κάνει την δουλειά που

θέλει. Με αυτόν τον τρόπο, έχουμε επιπρόσθετα και τα ακόλουθα πλεονεκτήματα:

1. Απόδοση. Το πλάνο εκτέλεσης του κάθε stored procedure αποθηκεύεται στην βάση μετά την πρώτη εκτέλεσή του, με αποτέλεσμα, να παραλείπονται τα βήματα του parsing και optimization που θα εκτελούνταν για κάθε εκτέλεση αν δεν κάναμε χρήση του stored procedure.
2. Διαχείριση κώδικα. Η λογική της βάσης μένει εντός της βάσης. Είναι ευκολότερα διαχειρίσιμη και λιγότερο επιρρεπής σε μη σωστή χρήση. Επίσης ο προγραμματιστής που θα χρειαστεί να κάνει εισαγωγές στην βάση θα αλληλεπιδρά με την βάση με μεγαλύτερη διαφάνεια, ευκολία, και επιπρόσθετα θα γράψει πολύ λιγότερες γραμμές κώδικα, πράγμα που μειώνει και τον χρόνο του debugging.
3. Προστασία από επιθέσεις τύπου SQL Injection. Επειδή κατά την εκτέλεση των stored procedures όλες οι παράμετροι (ανεξαρτήτως τύπου) πρέπει να είναι σε εισαγωγικά, δεν είναι δυνατή η εκτέλεση των sql injections.

Αυτό το οποίο χρειαζόμαστε για την εφαρμογή μας είναι πέντε stored procedures. Ένα για την εισαγωγή άρθρων, ένα για την εισαγωγή δημοπρασιών, ένα για την εισαγωγή πελάτη και εισαγωγή των λέξεων κλειδιών για τις οποίες ενδιαφέρεται, ένα για την απενεργοποίηση κάποιου πελάτη και ένα για την εκ νέου ενεργοποίησή του εφ' όσον χρειαστεί. Στο κάθε stored procedure θα πρέπει να εισάγονται ως παράμετροι όλα τα στοιχεία τα οποία σχετίζονται άμεσα ή έμμεσα με το βασικό στοιχεία. Όλα αυτά τα στοιχεία υπάρχουν στο αρχείο XML το οποίο παράγεται από το παραγωγικό υποσύστημα. Επιπρόσθετα το stored procedure θα πρέπει να ελέγχει πριν από κάθε εισαγωγή αν αυτό που επρόκειτο να εισαχθεί υπάρχει ήδη. Φυσικά η σειρά με την οποία θα γίνονται οι εισαγωγές στους επιμέρους πίνακες πρέπει να είναι τέτοια έτσι ώστε να είναι διαθέσιμα όλα τα στοιχεία τα οποία χρειάζονται για την εισαγωγή, όπως τα id των ξένων κλειδιών. Τέλος, τα stored procedure πρέπει να εξασφαλίζουν πως αν κάτι δεν πάει καλά κατά την διαδικασία της εισαγωγής/τροποποίησης να κάνουν rollback την βάση

στην κατάσταση πριν την εκτέλεση του stored procedure και να ενημερώσει τον κώδικα (επιστρέφοντας το μήνυμα λάθους) για το πρόβλημα το οποίο δημιουργήθηκε.

Έτσι, τα stored procedures τα οποία δημιουργήθηκαν είναι τα ακόλουθα:

A) sp_deactivateCompany

Το stored procedure sp_deactivateCompany χρησιμεύει στην απενεργοποίηση ενός πελάτη. Αυτό σημαίνει πως από την ημέρα απενεργοποίησης και έπειτα δεν θα λαμβάνει πλέον αποδελτίωση, όμως όλα τα αποκόμματά του θα συνεχίζουν να είναι διαθέσιμα για ορισμένο χρονικό διάστημα (πχ 20 ημέρες). Αυτή είναι και η χρησιμότητα αυτής της πληροφορίας. Δηλαδή, δίνει την δυνατότητα να σβηστούν όλα τα δεδομένα του πελάτη με το που θα επέλθει η περίοδος των 20 ημερών.

Στο stored procedure (βλ. Παράρτημα Δ, sp_deactivateCompany, σελ. 158) το @code είναι η μοναδική παράμετρος εισόδου και αντιστοιχεί στον κωδικό του πελάτη τον οποίο θέλουμε να απενεργοποιήσουμε. Τα @errMsg και @MyErrCode είναι παράμετροι εξόδου (συνοδεύονται από το keyword “OUT”). Σε αυτές τις παραμέτρους αποθηκεύεται το μήνυμα λάθους στην περίπτωση που η εντολή update δεν μπορέσει να εκτελεστεί επιτυχώς και ο κωδικός λάθους που θέλουμε να επιστρέφει στην κάθε περίπτωση.

Το ERROR_MESSAGE() είναι μία ενσωματωμένη συνάρτηση συστήματος της SQL η οποία επιστρέφει το μήνυμα λάθους. Το GETDATE() από την άλλη επιστρέφει την τρέχουσα ημερομηνία.

Στην SQL 2008 για πρώτη φορά εισάγεται η δυνατότητα χρήσης των blocks TRY, CATCH. Έτσι, αν συμβεί οποιοδήποτε λάθος εντός του block TRY, η εκτέλεσή του block TRY θα τερματιστεί και θα μεταβεί στο block CATCH.

B) sp_activateCompany

Το stored procedure sp_activateCompany (βλ. *Κώδικας 1: sp_activateCompany stored procedure*) χρησιμεύει στην ενεργοποίηση ενός πελάτη ο οποίος προηγουμένως είχε απενεργοποιηθεί.

```
-- =============================================

-- Author:        <Evelina Rimpapova>
-- Create date: <2011>
-- Description:    < Μέσα από αυτό το stored procedure μπορούμε να
-- ενεργοποιήσουμε ξανά έναν πελάτη. Αν στο πεδίο
-- contract_expiration_date του πίνακα Company υπάρχει δηλωμένη
-- ημερομηνία,τότε ο πελάτης έχει σταματήσει την συνδρομή του.
-- Η ημερομηνία δηλώνει το πότε σταμάτησε.
-- =============================================
CREATE PROCEDURE [dbo].[sp_activateCompany]
      @code nvarchar(50),

      @errMsg nvarchar(500) OUT,
      @myErrCode int OUT
AS
BEGIN
      SET NOCOUNT ON;
      BEGIN TRY
      update Company set contract_expiration_date = null where code = @code
      Select @errMsg = 'No error',@myErrCode = 0
      END TRY

      BEGIN CATCH
            SELECT @errMsg = ERROR_MESSAGE(), @myErrCode = 1
      END CATCH
END
      GO
```

Κώδικας 1: *sp_activateCompany stored procedure*

Γ) sp_insertKeyword

Το stored procedure sp_insertKeyword (βλ παράρτημα Δ, sp_insertKeyword, σελ. 160) εισάγει τις απαραίτητες πληροφορίες του πελάτη (πίνακας Company) και τις λέξεις κλειδιά (πίνακας Keyword) για τις οποίες ενδιαφέρεται (πίνακας InterestedIn), εφ' όσον αυτές οι πληροφορίες δεν έχουν καταχωρηθεί. Το sp_insertKeyword μπορεί να κληθεί είτε απευθείας είτε μέσα από άλλα stored procedure. Εξαιτίας αυτού, αξίζει να σχολιάσουμε τα ακόλουθα τμήματα του *Κώδικας 2: sp_insertKeyword stored procedure*:

Το @@TRANCOUNT είναι μια ενσωματωμένη συνάρτηση συστήματος της SQL η οποία επιστρέφει έναν αριθμό ο οποίος υποδηλώνει το πόσα είναι τα ενεργά transactions στο τρέχον session. Κατά την εκτέλεση ενός stored procedure όσα είναι τα ενεργά transactions κατά την έναρξη εκτέλεσής του, τόσα πρέπει να είναι και κατά τον τερματισμό του. Διαφορετικά σημαίνει πως κάποιο transaction δεν έκανε rollback ή commit.

Κάθε φορά που ξεκινάει ένα transaction με την εντολή BEGIN TRAN το @@trancount αυξάνεται αυτόματα κατά ένα. Με τις εντολές commit και rollback μειώνετε αυτόματα κατά ένα.

Στην περίπτωση που η sp_insertKeyword καλείτε απ' ευθείας (και όχι μέσω άλλου stored procedure), δεν θα υπάρχει κάποιο ενεργό transaction και άρα το @@TRANCOUNT θα είναι ίσο με μηδέν πράγμα που κάνει την πρώτη συνθήκη ελέγχου μη αληθής (βλ. *Κώδικας 2: sp_insertKeyword stored procedure*). Έτσι, δημιουργείτε το νέο transaction με όνομα myTran2. Επιπρόσθετα, καταχωρείται στην μεταβλητή @new η τιμή 1 που σημαίνει πως δημιουργήθηκε νέο transaction. Αν όλα τα ερωτήματα εκτελεστούν χωρίς πρόβλημα θα εκτελεστεί το "else" σκέλος της τελευταία συνθήκης ελέγχου του TRY block, όπου το transaction θα γίνει commit. Αν κάτι δεν εκτελεστεί σωστά, η εκτέλεση του κώδικα θα μεταφερθεί στο block CATCH, όπου θα εκτελεστεί η εντολή "ROLLBACK TRANSACTION myTran2". Τέλος, θα επιστραφεί το μήνυμα λάθους και ο κωδικός λάθους.

Στην δεύτερη περίπτωση, αν δηλαδή η sp_insertKeyword κληθεί μέσω άλλου stored procedure η οποία κατά την έναρξή της ξεκινάει νέο transaction, το @@TRANCOUNT κατά την κλήση της sp_insertKeyword θα έχει την τιμή ένα. Στην περίπτωσή μας τα sp_insertArticle και sp_insertTender, τα οποία θα δούμε στην συνέχεια, ξεκινάνε το transaction "myTran" πριν την εκτέλεση των εντολών. Έτσι, όταν κληθεί η sp_insertKeyword, το @@TRANCOUNT θα είναι μεγαλύτερο του μηδενός, και ίσο με ένα. Σε αυτή την περίπτωση δεν ξεκινάμε νέο transaction αλλά δημιουργούμε ένα savepoint του υπάρχοντος transaction μέσω της εντολής "SAVE TRAN myTran". Όμοια, μετά την εκτέλεση των εντολών, δημιουργούμε ακόμα ένα savepoint. Αν κάτι δεν εκτελέστηκε σωστά, μέσω του CATCH block κάνουμε ROLLBACK το transaction myTran και επιστρέφουμε το μήνυμα και τον κωδικό λάθους.

```
declare @new bit;
BEGIN TRY


      IF @@TRANCOUNT > 0
            BEGIN
                  SAVE TRAN myTran
                  set @new = 0
            END
      ELSE
            BEGIN
                  BEGIN TRAN myTran2
                  set @new = 1
            END
      BEGIN TRY

......

IF @new = 0 SAVE TRANSACTION myTran
      ELSE COMMIT TRANSACTION myTran2;
END TRY

BEGIN CATCH
      IF @new = 1
      BEGIN
            ROLLBACK TRANSACTION myTran2
      END
      ELSE
      BEGIN
            ROLLBACK TRANSACTION myTran
      END
      set @errMsg = ERROR_MESSAGE()
      set @myErrCode = 1
END CATCH
```

Κώδικας 2: *sp_insertKeyword stored procedure*

Δ) sp_insertArticle

Το stored procedure sp_insertArticle εισάγει τα άρθρα στην βάση (βλ παράρτημα Δ, sp_insertArticle, σελ. 163).

Ε) sp_insertTender

Το stored procedure sp_insertTender εισάγει της δημοπρασίες στην βάση (βλ παράρτημα Δ, sp_insertTender, σελ. 166).

4.5.3 Triggers

Με τα stored procedures εξασφαλίσαμε πως τα δεδομένα θα μπαίνουν με τρόπο που θα ακολουθεί την λογική της σχεδίασης της βάσης. Αυτό που όμως δεν έχει εξασφαλιστεί μέχρι στιγμής είναι ότι οι συνθήκες αυτές θα συνεχίζουν να ισχύουν μετά από διαγραφή δεδομένων. Αν για παράδειγμα μία εταιρεία έχει ένα χρήστη και διαγράψουμε αυτόν τον χρήστη, η εταιρεία δεν θα έχει πλέον κανένα χρήστη, πράγμα που παραβαίνει τους περιορισμούς που έχουμε θέσει. Ένα κείμενο είναι είτε άρθρο είτε δημοπρασία (υποχρεωτικά μόνο ένα από τα δύο). Αν διαγράψουμε ένα άρθρο, το κείμενο θα παραμείνει στον πίνακα Text και δεν θα αναφέρετε ούτε σε άρθρο ούτε σε δημοπρασία. Μέσα από αυτά τα δύο παραδείγματα είναι εμφανή η ανάγκη για την εισαγωγή των κατάλληλων triggers τα οποία θα εξασφαλίζουν τις συνθήκες που έχουμε περιγράψει. Για την ακρίβεια θέλουμε μετά από διαγραφή οποιοδήποτε ξένου κλειδιού (εκτός των company_id και keyword_id του πίνακα Clip) να διαγράφεται και η αντίστοιχη εγγραφή του πίνακα στον οποίο αναφέρεται. Έτσι θα καλυφθούν όλες οι ακόλουθες συνθήκες:

- Αν διαγραφεί άρθρο (βλ. *Κώδικας 3: Δημιουργία Trigger που αφορά τον πίνακα Article*), δημοπρασία (βλ. *Κώδικας 4: Δημιουργία Trigger που αφορά τον πίνακα Tender*) ή απόκομμα (βλ. *Κώδικας 5: Δημιουργία Trigger που αφορά τον πίνακα Clip*), θα διαγραφεί και το αντίστοιχο κείμενο με το οποίο συνδέονταν.

```
-- =============================================
-- Author:        <Evelina Rimpapova>
-- Create date: <Αύγουστος 2011>
-- Description:   <Διαγράφει τις εγγραφές του πίνακα Text
-- οι οποίες δεν αναφέρονται ως άρθρο ή δημοπρασία στους
-- πίανκες Article και Tender.>
-- =============================================
CREATE TRIGGER [dbo].[delOrphanTextRecords_Article]
   ON  [dbo].[Article]
   AFTER DELETE
AS
BEGIN
      SET NOCOUNT ON;

    delete from Text
    where id not in (Select text_id from Article)
    and id not in (select text_id from Tender)
END
GO
```

Κώδικας 3: *Δημιουργία Trigger που αφορά τον πίνακα Article*

```
-- =============================================
-- Author:        <Evelina Rimpapova>
-- Create date: <Αύγουστος 2011>
-- Description:   <Διαγράφει τις εγγραφές του πίνακα Tender οι
-- οποίες δεν αναφέρονται ως άρθρο ή δημοπρασία στους
-- πίανκες Article και Tender.>
-- =============================================
CREATE TRIGGER [dbo].[delOrphanTextRecords_Tender]
   ON  [dbo].[Tender]
   AFTER DELETE
AS
BEGIN
      SET NOCOUNT ON;

    delete from dbo.Text
    where id not in (Select text_id from dbo.Article)
    and id not in (select text_id from dbo.Tender)

END
GO
```

Κώδικας 4: *Δημιουργία Trigger που αφορά τον πίνακα Tender*

```
-- =============================================
-- Author:        <Evelina Rimpapova>
-- Create date: <Αύγουστος 2011>
-- Description:   <Διαγράφει τις εγγραφές του πίνακα Text
-- για τις οποίες δεν υπάρχει αναφορά στον πίνακα Clip.>
-- =============================================
CREATE TRIGGER [dbo].[delOrphanTextRecords_Clip]
   ON  [dbo].[Clip]
   AFTER DELETE
AS
BEGIN
      SET NOCOUNT ON;

    delete from dbo.Text
    where id not in (Select distinct text_id from dbo.Clip)

END
GO
```

Κώδικας 5: *Δημιουργία Trigger που αφορά τον πίνακα Clip*

- Αν διαγραφεί κάποιο κείμενο και η έκδοση στην οποία αναφερόταν δεν αναφέρεται σε άλλο κείμενο θα διαγραφεί και η έκδοση (βλ. *Κώδικας 6: Δημιουργία Trigger που αφορά τον πίνακα Text*).

```
-- =============================================
-- Author:        <Evelina Rimpapova>
-- Create date: <Αύγουστος 2011>
-- Description:   <Διαγράφει τις εγγραφές του πίνακα Text
-- για τις οποίες δεν υπάρχει αναφορά στον πίνακα Issue.>
-- =============================================
CREATE TRIGGER [dbo].[delOrphanIssueRecords_Text]
   ON  [dbo].[Text]
   AFTER DELETE
AS
BEGIN
      SET NOCOUNT ON;

    delete from dbo.Issue
    where id not in (Select distinct issue_id from dbo.Text)

END
GO
```

Κώδικας 6: *Δημιουργία Trigger που αφορά τον πίνακα Text*

- Αν διαγραφεί κάποια έκδοση και δεν υπάρχει άλλη έκδοση στην οποία να αναφέρεται κάποιο έντυπο, θα διαγραφεί και το έντυπο. (βλ. *Κώδικας 7: Δημιουργία Trigger που αφορά τον πίνακα Issue*)

```
-- =============================================
-- Author:        <Evelina Rimpapova>
-- Create date: <Αύγουστος 2011>
-- Description:   <Διαγράφει τις εγγραφές του πίνακα PressMedia
-- για τις οποίες δεν υπάρχει αναφορά στον πίνακα Issue.>
-- =============================================
CREATE TRIGGER [dbo].[delOrphanPressMediaRecords_Issue]

   ON  [dbo].[Issue]
   AFTER DELETE
AS
BEGIN
      SET NOCOUNT ON;

    delete from dbo.PressMedia
    where id not in (Select distinct id from dbo.Issue)

END
GO
```

Κώδικας 7: *Δημιουργία Trigger που αφορά τον πίνακα Issue*

- Αν κάποιος πελάτης σταματήσει να ενδιαφέρεται για κάποια λέξη κλειδί και εφ' όσον αποφασιστεί να γίνει διαγραφή αυτής από την βάση, θα πρέπει να διαγραφεί η αναφορά από τον πίνακα InterestedIn. Αν κανένας άλλος πελάτης δεν ενδιαφέρεται για αυτήν την λέξη κλειδί, η λέξη κλειδί θα πρέπει να διαγραφεί. Επιπρόσθετα αν αυτός ο πελάτης δεν έχει άλλες λέξεις για τις οποίες ενδιαφέρεται, πρέπει και ο πελάτης να διαγραφεί. (βλ. *Κώδικας 8: Δημιουργία Trigger που αφορά τον πίνακα InterestedIn*).

```
-- =============================================
-- Author:        <Evelina Rimpapova>
-- Create date: <Αύγουστος 2011>
-- Description:   <Διαγράφει τις εγγραφές των πινάκων Company
-- και Keyword για τις οποίες δεν υπάρχει αναφορά
-- στον πίνακα InterestedIn.>
-- =============================================
CREATE TRIGGER [dbo].[delOrphanKeywordRecords_InterestedIn]

   ON  [dbo].[InterestedIn]
   AFTER DELETE
AS
BEGIN
      SET NOCOUNT ON;

    delete from dbo.Keyword
    where id not in (Select distinct keyword_id from dbo.InterestedIn)

    delete from dbo.Company
    where id not in (select distinct company_id from dbo.InterestedIn)

END
GO
```

Κώδικας 8: *Δημιουργία Trigger που αφορά τον πίνακα InterestedIn*

- Αν διαγραφεί ένας χρήστης μιας εταιρείας και δεν υπάρχει άλλος χρήστης για αυτήν την εταιρεία, πρέπει και η εταιρεία να διαγραφεί. (βλ. *Κώδικας 9: Δημιουργία Trigger που αφορά τον πίνακα LoginUser*).

```
-- =============================================
-- Author:      <Evelina Rimpapova>
-- Create date: <Αύγουστος 2011>
-- Description: <Διαγράφει τις εγγραφές του πίνακα
-- Company για τις οποίες δεν υπάρχει αναφορά LoginUser.>
-- =============================================
CREATE TRIGGER [dbo].[delOrphanTextRecords_LoginUser]
   ON  [dbo].[LoginUser]
   AFTER DELETE
AS
BEGIN
    SET NOCOUNT ON;

    delete from dbo.Company
    where id not in (Select distinct id from dbo.LoginUser)

END
GO
```

Κώδικας 9: *Δημιουργία Trigger που αφορά τον πίνακα LoginUser*

Εξαίρεση έχουμε μόνο στην περίπτωση που διαγράφουμε ένα απόκομμα. Όπως αναφέραμε, θα διαγραφεί το κείμενο στο οποίο αναφέρεται αλλά δεν θα

διαγραφεί η λέξη κλειδί στην οποία αναφέρεται ούτε ο πελάτης που έλαβε αυτό το απόκομμα, ακόμα και αν αυτό ήταν το μοναδικό απόκομμα. Αυτό γιατί μπορεί ένας πελάτης να είναι καταχωρημένος στην βάση και να μην έχει αποκόμματα και ταυτόχρονα πρέπει να υπάρχουν καταχωρημένες οι λέξεις κλειδιά για τις οποίες ενδιαφέρεται.

Τέλος, όταν διαγραφεί ένα κύριο κλειδί το οποίο έχει αναφορές σε άλλους πίνακες θα πρέπει να διαγράφονται και οι αναφορές τους (on delete cascade). Οι on delete cascade εντολές φαίνονται στο παράρτημα Δ, Δημιουργία πινάκων με SQL εντολές. 147. Για τον πίνακα Clip για παράδειγμα, ισχύουν οι ακόλουθες εντολές:

```
ALTER TABLE [dbo].[Clip]  WITH CHECK ADD  CONSTRAINT [FK_Clip_company_id]
FOREIGN KEY([company_id])
REFERENCES [dbo].[Company] ([id])
ON DELETE CASCADE
GO

ALTER TABLE [dbo].[Clip]  WITH CHECK ADD  CONSTRAINT [FK_Clip_keyword_id]
FOREIGN KEY([keyword_id])
REFERENCES [dbo].[Keyword] ([id])
ON DELETE CASCADE
GO

ALTER TABLE [dbo].[Clip]  WITH CHECK ADD  CONSTRAINT [FK_Clip_text_id]
FOREIGN KEY([text_id])
REFERENCES [dbo].[Text] ([id])
ON DELETE CASCADE
GO
```

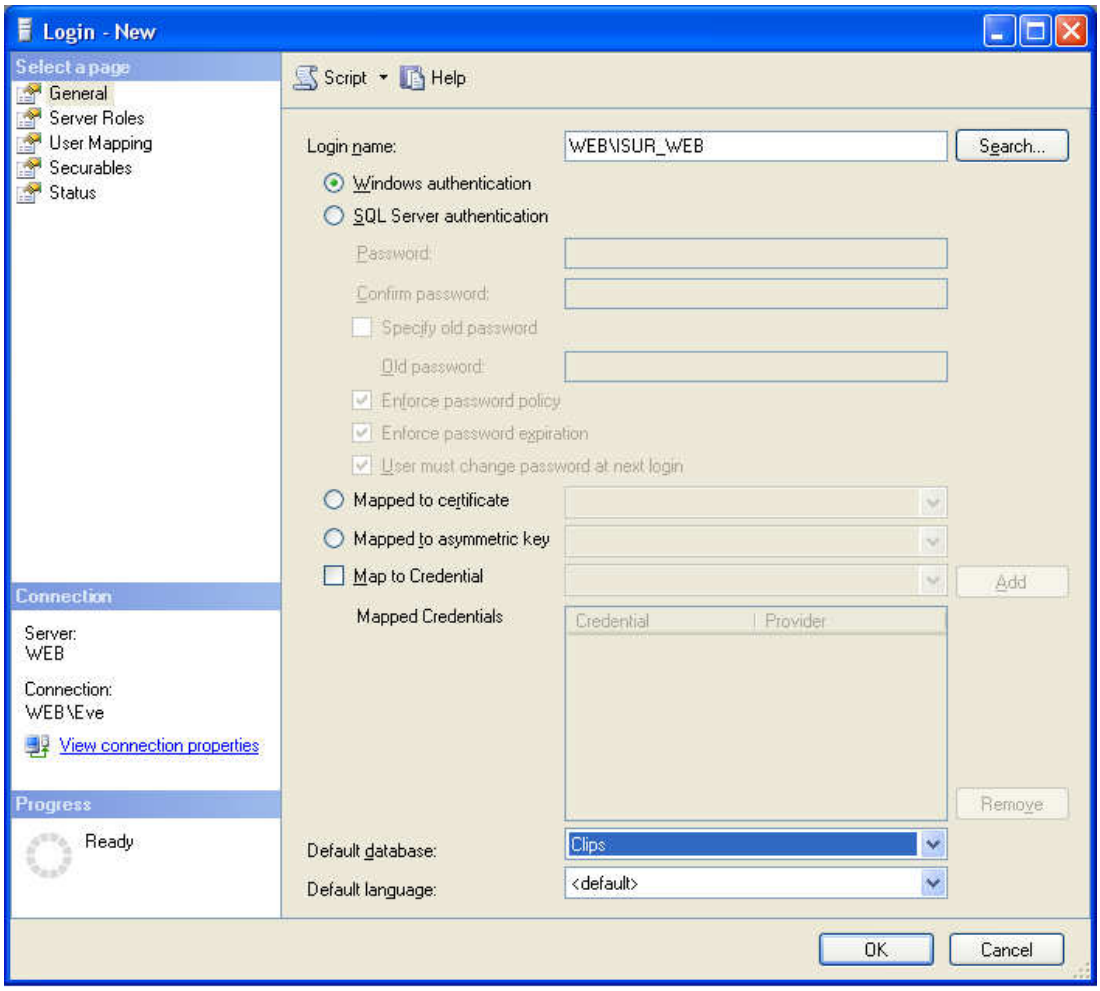
Κώδικας 10: *On delete cascade εντολές που αφορούν τον πίνακα Clip*

4.5.4 Windows IUSR account και Permissions

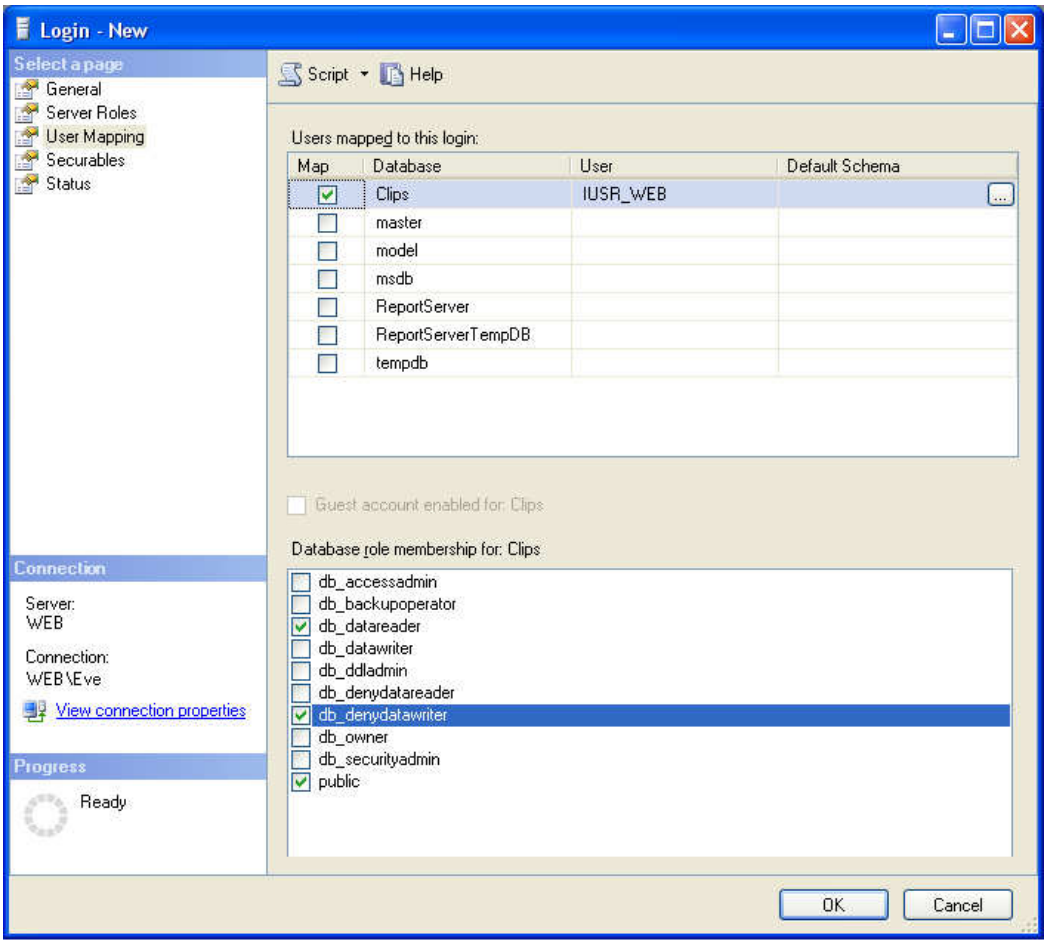
Η βάση που υλοποιήσαμε θα είναι αυτή η οποία θα κρατάει τις απαιτούμενες πληροφορίες των αποκομμάτων έτσι ώστε οι πελάτες μέσω του site της εταιρείας να μπορούν να δουν την αποδελτίωσή τους. Οι χρήστες εισάγοντας ένα username και password θα αποκτούν πρόσβαση σε αυτές τις πληροφορίες από την στιγμή που λαμβάνουν αποδελτίωση. Τα στοιχεία αυτά χρειάζονται και για να διαπιστώσουμε το ποιος είναι ο πελάτης. Πριν όμως ο χρήστης φτάσει στο σημείο του Login, πρώτα έχει επισκεφθεί τον ιστότοπο της εταιρείας αποδελτίωσης. Πρόσβαση στο site έχουν οι χρήστες χάρις στον IIS (Internet Information Server).

Στον IIS ρυθμίζεται με ποιόν τρόπο οι χρήστες συνδέονται στο site. Ο πιο κοινός τρόπος σύνδεσης είναι μέσο ανώνυμης πρόσβασης (anonymous authentication). Δηλαδή ο χρήστης δεν εισάγει κάποιο username και password για να επισκεφθεί μία ιστοσελίδα. Οι χρήστες που συνδέονται με αυτό τον τρόπο σε ένα Server ουσιαστικά συνδέονται μέσω του IUSR_όνομαServer λογαριασμού ο οποίο υπάρχει εγκατεστημένος από Default με την εγκατάσταση του IIS. Ο λογαριασμός IUSR_ WEB (έστω ότι ο Server μας ονομάζεται WEB) ανήκει στο group Guests. Για το παράδειγμα μας θα χρησιμοποιήσουμε αυτόν τον λογαριασμό για τις συνδέσεις που απαιτούνται από τον κώδικα της ιστοσελίδας προς την βάση (θα μπορούσε να είναι οποιοσδήποτε άλλος λογαριασμός ο οποίος υπάρχει στον Server και θα συνδέεται με Windows authentication στη βάση ή κάποιος λογαριασμός που θα υπάρχει μόνο στην SQL και θα συνδέεται με SQL authentication).

Στο Management Studio της SQL θα πρέπει να δημιουργήσουμε ένα Server Login (Object explorer: WEB → Security (δεξί click) → New → login) για τον χρήστη IUSR_WEB. Αυτό θα δώσει την δυνατότητα σε αυτόν τον λογαριασμό να έχει πρόσβαση στον SQL Server. Κατά τη διαδικασία της δημιουργίας του Server login, δίνουμε πρόσβαση του χρήστη στην βάση Clips όπως και ρόλους db_datareader και db_denydatawriter μέσα από την καρτέλα User Mapping. Έτσι, ο χρήστης θα έχει το δικαίωμα να βλέπει την βάση Clip, να διαβάζει δεδομένα από αυτήν αλλά δεν θα έχει το δικαίωμα να γράφει σε αυτήν (ο χρήστης δηλαδή δεν θα μπορεί να εκτελέσει εντολές Insert, Update και Delete). Στις εικόνες: *Εικόνα 18: Δικαιώματα χρήστη IUSR_WEB, σελίδα General Εικόνα 19: Δικαιώματα χρήστη IUSR_WEB, σελίδα User Mapping*, *Εικόνα 20: Δικαιώματα εκτέλεσης των stored procedures για τον χρήστη IUSR_WEB* φαίνονται οι απαραίτητες ρυθμίσεις.

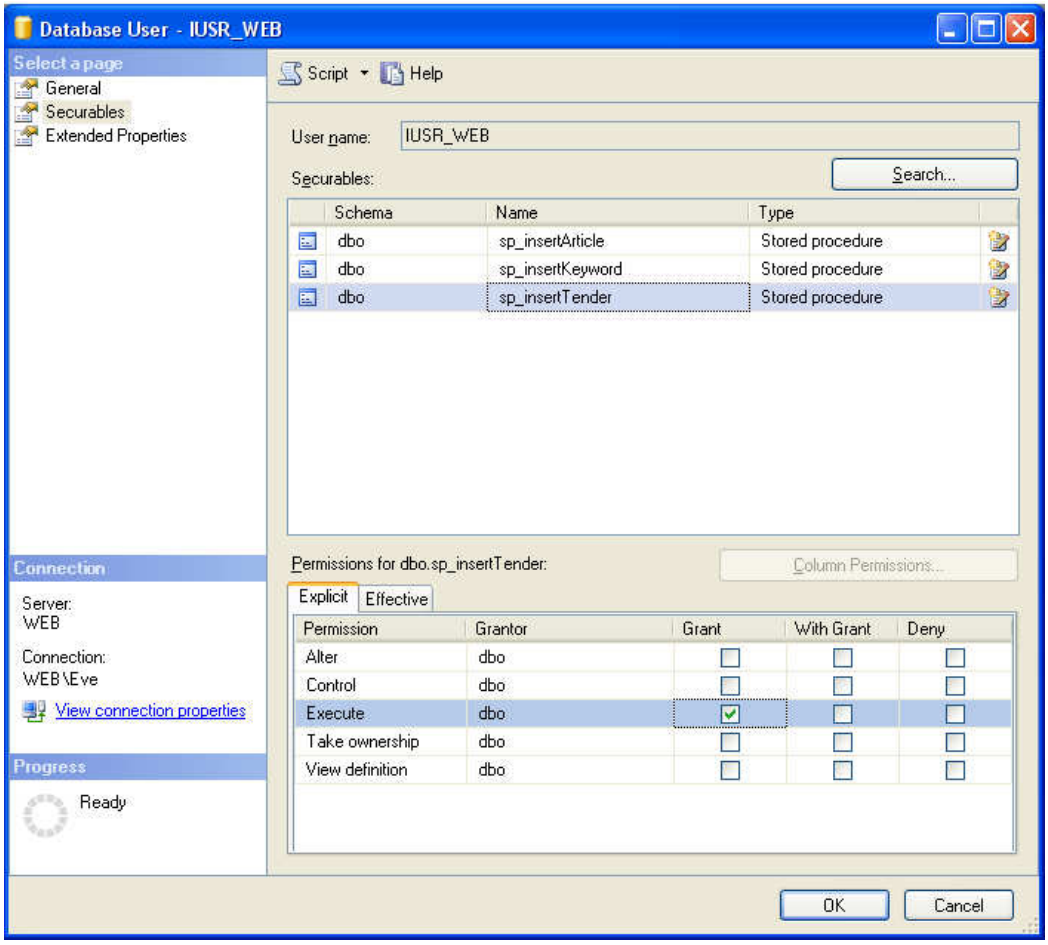


Εικόνα 18: *Δικαιώματα χρήστη IUSR_WEB, σελίδα General*



Εικόνα 19: *Δικαιώματα χρήστη IUSR_WEB, σελίδα User Mapping*

Τέλος, αυτό που μένει είναι να δώσουμε δικαιώματα εκτέλεσης για τα stored procedures που έχουν δημιουργηθεί για τον χρήστη IUSR_WEB (Object Explorer: Web → Databases → Clips → Security → WEB\IUSR_WEB (δεξί click) → Properties) έτσι ώστε να μπορεί να διαχειριστεί τα δεδομένα της βάσης με τον τρόπο που έχουμε καθορίσει εμείς.



Εικόνα 20: *Δικαιώματα εκτέλεσης των stored procedures για τον χρήστη IUSR_WEB*

Κεφάλαιο 5: Διασύνδεση των δύο υποσυστημάτων – λειτουργικές προδιαγραφές

5.1 Εισαγωγή

Σε αυτό το κεφάλαιο θα αναλύσουμε τις προδιαγραφές της εφαρμογής όπως και το ποιοι είναι οι περιορισμοί, οι ιδιαιτερότητες, τα προβλήματα και οι ανάγκες που την διαμόρφωσαν. Είναι αξιοσημείωτο το πώς οι ανάγκες και οι προδιαγραφές εξελίχθηκαν μέσα στο χρόνο. Η εφαρμογή η οποία περιγράφουμε έφτασε στην τελική της μορφή μετά από πολλές μεγάλες ενημερώσεις. Θα προσπαθήσουμε να δώσουμε την λογική και τα προβλήματα που οδήγησαν σε κάποιες προδιαγραφές, μιας και είναι προβλήματα τα οποία ήταν δύσκολο να προνοήσουμε εξ αρχής και προέκυψαν από την εμπειρία και την χρήση της εφαρμογής.

5.2 Λειτουργικές προδιαγραφές.

Η εφαρμογή αυτή καλύπτει τις ακόλουθες λειτουργικές ανάγκες στην εταιρεία αποδελτίωσης εντύπων:

1. Διάθεση των αποκομμάτων σε DjVu μορφή.

Το μοναδικό σκέλος της εφαρμογής το οποίο δεν σχετίζεται με βάσεις δεδομένων είναι η μετατροπή (convert) του PDF αρχείου, σε αρχείο τύπου DjVu. Για να γίνει η μετατροπή αυτή απαιτείται η χρήση των κατάλληλων βιβλιοθηκών. Έτσι, ενώ αγοράστηκε μία βιβλιοθήκη για αυτόν το σκοπό, αποδείχθηκε πως ήταν προβληματική κατά την μετατροπή κάποιων PDF αρχείων, για άγνωστη μέχρι στιγμής αιτία. Δηλαδή, όταν δεχόταν κάποια συγκεκριμένα αρχεία PDF δεν μπορούσε να δημιουργήσει το αντίστοιχο αρχείο σε μορφή DjVu, βγάζοντας μήνυμα λάθους κατά την λειτουργία της. Για να λύσουμε αυτό το πρόβλημα, έγινε αναζήτηση για άλλη βιβλιοθήκη. Η βιβλιοθήκη που βρέθηκε, η οποία ήταν και δωρεάν, δεν αντιμετώπιζε το πρόβλημα της πρώτης, όμως δεν διέθετε τόσες πολλές δυνατότητες παραμετροποίησης όσο η πρώτη, με αποτέλεσμα το παραγόμενο DjVu

αρχείο να έχει μεγαλύτερο μέγεθος από το θεμιτό. Έτσι, αποφασίστηκε να χρησιμοποιηθεί η πρώτη, η εμπορική βιβλιοθήκη, αλλά στις λίγες περιπτώσεις που το convert αποτύχανε (περίπου 1-15 στα 2000 αρχεία) η εφαρμογή να χρησιμοποιεί την δεύτερη βιβλιοθήκη. Είναι κρίσιμο όλα, ανεξαιρέτως, τα αρχεία να υπάρχουν και στα δύο format διαθέσιμα στο πελάτη, και άρα είναι κρίσιμο για την εφαρμογή να μην ανεβάζει απόκομμα αν αυτή η προϋπόθεση δεν τηρείται.

2. Δημοσιοποίηση των αποκομμάτων που έχουν παραχθεί από το παραγωγικό υποσύστημα καταχωρώντας τα στο DBMS από το οποίο ο ιστότοπος της εταιρείας αντλεί δεδομένα.

Το δεύτερο και σημαντικότερο σκέλος της εφαρμογής έχει να κάνει με την εισαγωγή των δεδομένων που υπάρχουν στα XML αρχεία, στην βάση δεδομένων που υποστηρίζει τον ιστότοπο της εταιρείας, έτσι ώστε να είναι διαθέσιμα τα αποκόμματα στους χρήστες. Αυτή η διαδικασία αποτελείται από τις ακόλουθες περιπτώσεις:

Στην περίπτωση που το XML αρχείο περιέχει πληροφορίες οι οποίες δεν υπάρχουν στην βάση οι πληροφορίες αυτές θα πρέπει να εισαχθούν. Στην περίπτωση που το XML αρχείο περιέχει πληροφορίες οι οποίες έχουν τροποποιηθεί με βάση τις ήδη καταχωρημένες πληροφορίες θα πρέπει να τροποποιηθούν οι αντίστοιχες εγγραφές. Για παράδειγμα, αν αλλάξει η νομική μορφή μιας εταιρείας και από Ο.Ε γίνει Α.Ε και άρα αλλάξει και η επωνυμία της, αυτή η αλλαγή πρέπει να εφαρμοστεί. Άλλα πεδία τα οποία επιδέχονται τέτοιου είδους αλλαγές είναι το όνομα της λέξης κλειδί και το όνομα και ο τύπος ενός εντύπου. Μαζί με την καταχώρηση κάθε νέου αποκόμματος, θα πρέπει να μεταφέρονται τα αρχεία PDF, DJVU και XML στους αντίστοιχους φακέλους έτσι ώστε να είναι προσβάσιμα από τους πελάτες. Για την ακρίβεια, το XML αρχείο κρατιέται καθαρά για λόγους αρχειοθέτησης, κάτι το οποίο δεν αποτελεί πρόβλημα λόγω του μικρού του μεγέθους. Δηλαδή, από την στιγμή που οι πληροφορίες του XML εισαχθούν στην βάση, το αρχείο XML δεν χρησιμεύει σε κάτι.

3. Παρακολούθηση των αλλαγών (προσθήκες, τροποποιήσεις) που συμβαίνουν στο DBMS του παραγωγικού υποσυστήματος και εφαρμογή αυτών στο DBMS του ιστότοπου της εταιρείας.

Ένα από τα μεγαλύτερα προβλήματα ήταν ότι δεν υπήρχε τρόπος συγχρονισμού των βάσεων. Έτσι, δεν υπήρχε τρόπος η βάση του υποσυστήματος δημοσίευσης να ενημερώνεται για το αν κάποιος πελάτης διακόπηκε. Το σύστημα στέλνει XML και PDF αρχείο ανά απόκομμα αν και μόνο αν ο πελάτης είναι ενεργός. Το να διαγράφεται κάποιος πελάτης αυτόματα από την βάση του συστήματος δημοσίευσης στην περίπτωση που κάποιος λογαριασμός δεν έπαιρνε απόκομμα για πολύ μεγάλο χρονικό διάστημα, δεν μπορούσε να εφαρμοστεί, γιατί υπάρχουν λογαριασμοί που παίρνουν αποκόμματα περιοδικά (πχ μια φορά κάθε χρόνο, για διάφορα ετήσια event, όπως η Διεθνής Έκθεση Θεσσαλονίκης). Γενικά, αυτός ο τρόπος δεν θα ήταν αξιόπιστος. Η πιο εύκολη και βολική λύση ήταν να προσθέσουμε στην εφαρμογή την δυνατότητα να συγχρονίζει τις δύο βάσεις με προγραμματισμένο τρόπο. Μία φορά στο τέλος της ημέρας, θα τρέχουν ερωτήματα προς την βάση του παραγωγικού υποσυστήματος και θα ενημερώνει την βάση του υποσυστήματος δημοσίευσης. Οι ενημερώσεις αυτές αφορούν την δημιουργία νέων πελατών, την προσθήκη νέων λέξεων κλειδιών, την απενεργοποίηση, και την εκ νέου ενεργοποίηση πελάτη.

Κατά την έναρξη ενός πελάτη και κατά την δημιουργία ενός χρήστη, υποχρεωτικά πρέπει να συμπληρωθούν και τα στοιχεία εισόδου (username, password) του χρήστη, τα οποία όμως δεν είναι γνωστά. Η σύμβαση που θα ακολουθηθεί είναι οτι ο νέος χρήστης, του νέου αυτού πελάτη που δημιουργείται, να έχει username και password ίδιο με την επωνυμία του (το οποίο δίνεται στο XML).

Επίσης, σε αυτή την ηλεκτρονική διεύθυνση, η εφαρμογή θα πρέπει να στέλνει email ενημερώνοντας τον πελάτη ότι ο λογαριασμός του έχει ενεργοποιηθεί και τα βήματα τα οποία πρέπει να κάνει για να αλλάξει τους κωδικούς του. Δηλαδή, ο πελάτης θα πρέπει κατά το πρώτο Login να εισάγει το username και password τα οποία αναγράφονται στο email και αφού συνδεθεί να δημιουργήσει τα νέα στοιχεία εισόδου της επιλογής του.

5.3 Ειδικές απαιτήσεις.

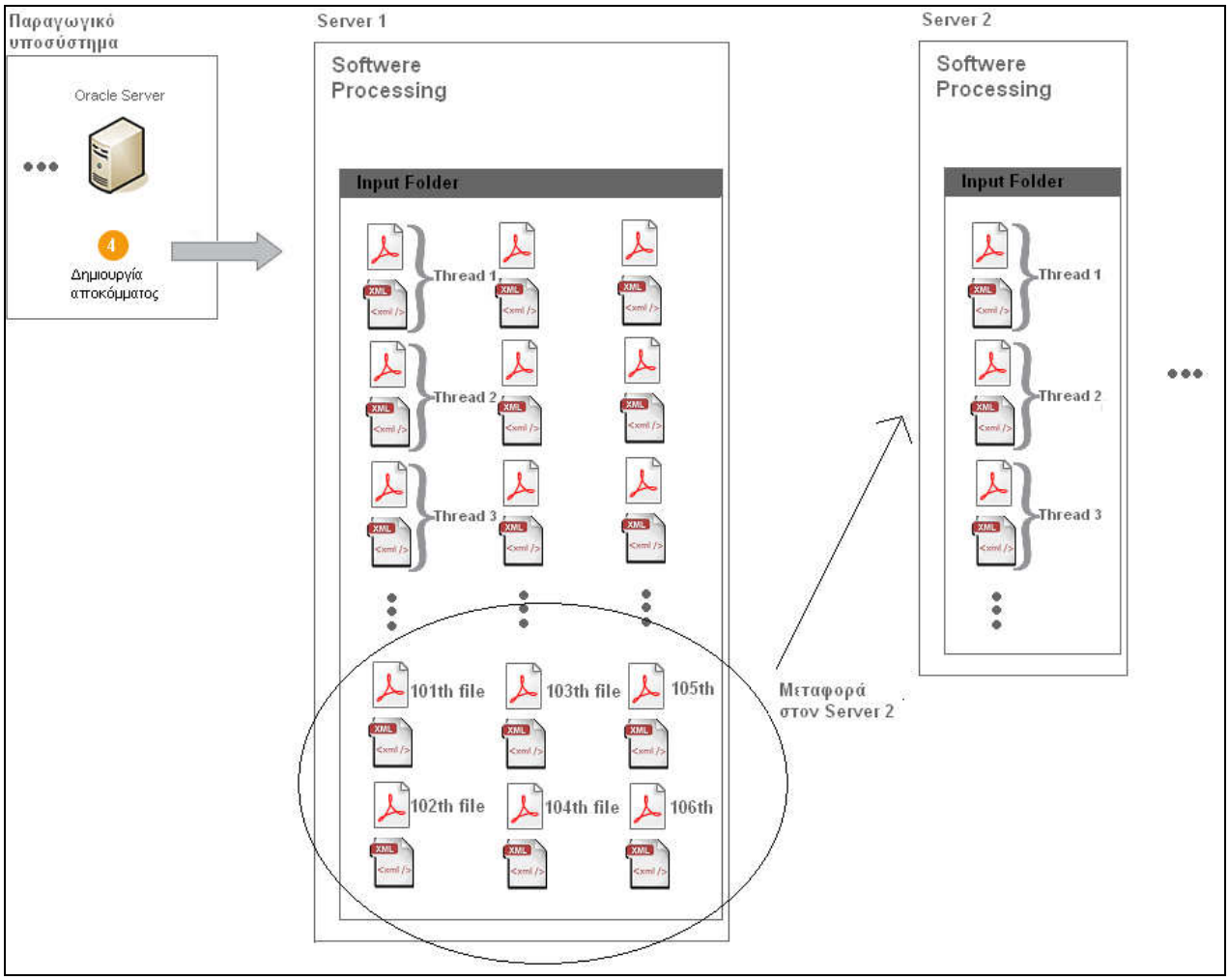
Προκειμένου να πετύχει η εφαρμογή στο μέγιστο δυνατό το σκοπό της απαιτούνται οι ακόλουθες ειδικές απαιτήσεις.

1. Η ταχύτητα είναι κάτι πολύ βασικό για την εφαρμογή. Η μετατροπή των PDF αρχείων σε DjVu είναι κάτι χρονοβόρο (από 8'' μέχρι 3'). Αυτό δημιουργεί ένα bottleneck στην παραγωγική διαδικασία. Η ροή των PDF και XML είναι μεγαλύτερη από αυτή που η εφαρμογή «προλαβαίνει» να επεξεργαστεί. Αυτό μπορούσε να οδηγήσει ακόμα και σε καθυστέρηση 2-3 ωρών. Το απόκομμα δηλαδή μπορεί να έχει παραχθεί από της 9:00 και να δημοσιεύονταν από την εφαρμογή στις 12:00 λόγω του αυξημένου όγκου αποκομμάτων. Η λύση που εφαρμόστηκε για να λυθεί αυτό το πρόβλημα είναι να γίνεται ταυτόχρονα από διαφορετικά threads η μετατροπή και η εισαγωγή των αρχείων. Κάθε ένα thread θα μετατρέπει ένα PDF σε DjVu και θα εισάγει τα δεδομένα του XML. Τρέχοντας πολλαπλά threads, θα γίνονται πολλαπλά convert ταυτόχρονα. Το πόσα thread θα έπρεπε να τρέχουν ταυτόχρονα δεν υπήρχε τρόπος να γίνει γνωστό εκ των προτέρων επειδή είναι κάτι το οποίο εξαρτάται από το hardware του server και μόνο πειραματικά θα μπορούσε να διαπιστωθεί. Επίσης, έπρεπε να ληφθεί υπόψη ότι ο server στον οποίο θα τρέχει η εφαρμογή, μελλοντικά θα μπορούσε να αντικατασταθεί με κάποιον πιο δυνατό και κατά επέκταση, θα μπορούσαν να τρέξουν περισσότερα threads. Άρα, καλόν θα ήταν, ο αριθμός των threads που θα τρέχει η εφαρμογή, να είναι εύκολα παραμετροποιήσιμος.

Ακόμα και μετά την πιο πάνω υλοποίηση, και πάλι υπήρχαν στιγμές όπου η εφαρμογή δεν μπορούσε να ανταποκριθεί στον φόρτο. Θεωρούμε πως μια καθυστέρηση της τάξης των 15' είναι δεκτή, ενώ μεγαλύτερη αποτελεί πρόβλημα. Δηλαδή δεν πρέπει να περνάνε πάνω από 15' από την ώρα που θα έχουν δημιουργηθεί τα PDF,XML αρχεία μέχρι την ώρα που θα έχουν ανέβει στην βάση δημοσίευσης. Προφανώς η χρήση παράλληλων threads δεν ήταν αρκετή. Επιπρόσθετα, μία από τις λύσεις που θα μπορούσε να δοθεί είναι η αντικατάσταση του server με δυνατότερο, αλλά και πάλι, αυτό που πρώτα πρέπει να σκεφτούμε είναι το τι μπορούμε να κάνουμε με τον ήδη υπάρχοντα εξοπλισμό. Έτσι εφαρμόστηκε η ακόλουθη ιδέα: Η εφαρμογή να

εγκατασταθεί σε πολλούς servers ταυτόχρονα. Σε αυτή την περίπτωση θα έπρεπε να εξασφαλιστεί πως δεν υπάρχει περίπτωση δύο από τις εφαρμογές να πάρουν ή να προσπαθήσουν να εισάγουν από κοινού το ίδιο αρχείο. Ένα δεύτερο πρόβλημα είναι ότι το παραγωγικό υποσύστημα στέλνει τα αρχεία των αποκομμάτων μόνο σε έναν φάκελο και δεν μπορεί να ρυθμιστεί έτσι ώστε να τα διαμοιράζει σε διαφορετικές διευθύνσεις φακέλων.

Τελικά η λύση που εφαρμόστηκε είναι η εξής: Έστω ότι για να καλύψουμε όλο τον φόρτο χρειάζεται να τρέχουν τρεις εφαρμογές (ίδιες μεταξύ τους), σε διαφορετικά μηχανήματα. Η κάθε εφαρμογή ελέγχει συγκεκριμένο φάκελο από τον οποίο αντλεί την κάθε δυάδα από Pdf και XML προς επεξεργασία. Η μία από αυτές, έστω η πρώτη, θα παρακολουθεί τον φάκελο όπου καταλήγουν τα αρχεία Pdf και XML από το παραγωγικό υποσύστημα. Η εφαρμογή αυτή θα λειτουργεί ανεξάρτητη από τις άλλες δύο, μετατρέποντας και εισάγοντας τα αποκόμματα. Αν όμως ο αριθμός των αποκομμάτων που «περιμένουν» να επεξεργαστούν από την εφαρμογή βρισκόμενα στον φάκελο, ξεπερνά για παράδειγμα τα 100, τότε θα μεταφέρει τα αποκόμματα τα οποία μπήκαν τελευταία στον φάκελο, στον φάκελο από τον οποίο αντλεί αποκόμματα η δεύτερη εφαρμογή (βλ. *Εικόνα 21: Μεταφορά αρχείων για την επίτευξη παράλληλης λειτουργίας της ίδιας εφαρμογής σε πολλούς Servers*. Η δεύτερη εφαρμογή με την σειρά της θα κάνει την ίδια διεργασία. Δηλαδή αν έχει πολλές δυάδες αρχείων προς αναμονή θα τα στείλει στην τρίτη εφαρμογή. Η τρίτη εφαρμογή με την σειρά της θα μεταφέρει τις δυάδες που «περιμένουν» στην πρώτη εφαρμογή, ολοκληρώνοντας τον κύκλο. Έτσι, εξασφαλίζεται ο ίσος καταμερισμός των αρχείων στο μέγιστο δυνατό και σύμφωνα με τις δυνατότητες του hardware. Αν δηλαδή η δεύτερη εφαρμογή τρέχει σε πιο γρήγορο μηχάνημα, τότε θα κρατάει μεγαλύτερο αριθμό αρχείων (πχ 200 αντί 100) προς επεξεργασία.



Εικόνα 21: *Μεταφορά αρχείων για την επίτευξη παράλληλης λειτουργίας της ίδιας εφαρμογής σε πολλούς Servers*

2. Παραμετροποίηση. Πέρα από τον αριθμό των threads υπάρχουν και πολλά ακόμα στοιχεία τα οποία αν και στατικά, υπάρχει η ανάγκη να είναι εύκολα παραμετροποιήσημα. Αυτά τα στοιχεία είναι οι διευθύνσεις του δίσκου όπου αποθηκεύονται τα αρχεία προς επεξεργασία (input path), η διεύθυνση όπου αποθηκεύονται τελικά τα αποκόμματα του πελάτη (output path), η διεύθυνση όπου αποθηκεύονται τα αρχεία για τα οποία κάτι δεν πήγε καλά κατά την επεξεργασία τους (error path) και οι διευθύνσεις των φακέλων εγκατάστασης των εξωτερικών εφαρμογών που χρησιμοποιούνται για το convert. Έτσι προκύπτει η ανάγκη του configuration file (αρχείο αποθήκευσης ρυθμίσεων) το οποίο θα αναλύσουμε στο επόμενο κεφάλαιο. Έτσι, μπορούμε με ευκολία να αλλάξουμε οποιαδήποτε από τις πιο πάνω μεταβλητές χωρίς να πειράξουμε τον κώδικα.

3. Log Files. Ειδικά κατά την ανάπτυξη της εφαρμογής αλλά και κατά τον πρώτο καιρό χρήσης, φάνηκε πολύ χρήσιμη η δυνατότητα της εφαρμογής να καταγράφει σε log file την λειτουργία της. Έτσι ήταν πιο εύκολο και γρήγορο να βρεθεί τι πήγε στραβά χωρίς να χρειάζεται να γίνει debugging.

4. Πολύ σημαντικό είναι και να εξασφαλιστεί ότι δεν θα γίνει commit στην βάση αν όλα δεν έχουν γίνει όπως θα έπρεπε (convert, μεταφορά αρχείων, εισαγωγή αποκόμματος) ακόμα και όταν σταματάμε την εφαρμογή. Δεν θα πρέπει δηλαδή να υπάρχει περίπτωση να έχουν μεταφερθεί τα αρχεία αλλά να μην έχει ενημερωθεί η βάση, και το αντίστροφο. Επίσης, στην περίπτωση λάθους, πρέπει να αποστέλλεται ειδοποίηση (email) στον αρμόδιο (πχ άτομο από την μηχανοργάνωση), προκειμένου να ενημερωθεί άμεσα ότι κάποιο αρχείο αντιμετώπισε κάποιο πρόβλημα, για να προβεί στην διόρθωσή του. Αυτό το οποίο ενδιαφέρει είναι η εφαρμογή να λειτουργεί χωρίς πρόβλημα, να μην χρειάζεται καθημερινό έλεγχο, και αν κάτι πάει στραβά, να αποστέλλει email στον αρμόδιο. Δηλαδή εξασφάλιση πλήρους αυτοματοποίησης.

5. Μία προβληματική κατάσταση, η οποία διαπιστώθηκε στην πορεία, προέκυπτε όταν δεν υπάρχει μεγάλη ουρά στα αποκόμματα. Η εφαρμογή, έτσι όπως έχει σχεδιαστεί, διαβάζει πολλά αρχεία παράλληλα. Στην περίπτωση που στην ουρά έχουμε λιγότερα αρχεία από τον αριθμό των threads, δηλαδή δεν έχουμε κανένα αρχείο που να περιμένει και έχουμε ελεύθερα threads (δλδ threads τα οποία δεν επεξεργάζονται κάποιο αρχείο), αυτό σημαίνει πως το επόμενο αρχείο το οποίο θα έρθει θα επεξεργαστεί άμεσα. Το πρόβλημα είναι ότι στην πλειοψηφία τέτοιων περιπτώσεων, η επεξεργασία του αρχείου ξεκινούσε πριν ολοκληρωθεί η δημιουργία και των δύο αρχείων (PDF και XML) από το υποσύστημα παραγωγής. Αυτό συνέβαινε κατά κύριο λόγο στις περιπτώσεις που το PDF έχει αρκετά μεγάλο μέγεθος και άρα αργούσε η δημιουργία του. Μέχρι αυτή να τελειώσει, το αρχείο πρακτικά δεν υπάρχει, ενώ το XML έχει είδη δημιουργηθεί. Για την αποφυγή αυτού του προβλήματος, η επόμενη έκδοση της εφαρμογής σχεδιάστηκε έτσι ώστε να αναζητά τις δυάδες αρχείων με βάση των αρχείων PDF και όχι με βάση των αρχείων XML. Δηλαδή θα ελέγχει τα περιεχόμενα του input φακέλου και θα θεωρεί ότι υπάρχει κάτι προς επεξεργασία μόνο αν υπάρχει PDF αρχείο εντός του φακέλου. Συγκεκριμένα, ενώ στην αρχική έκδοση η εφαρμογή έπαιρνε την λίστα περιεχομένων της διεύθυνσης εισόδου

(input path) μέσω XML φίλτρου (δηλαδή έφερνε μια λίστα περιεχομένων μόνο για τα XML αρχεία), στην επόμενη έκδοση το φίλτρο ήταν για PDF αρχεία, δηλαδή η λίστα περιεχομένων αφορούσε μόνο τα PDF αρχεία). Έτσι δεν υπήρχε πια το πρόβλημα γιατί τα PDF αρχεία έχουν πάντα μεγαλύτερο μέγεθος και αυτό καθιστά σχεδόν σίγουρο ότι όταν θα υπάρχει το PDF θα υπάρχει και το αντίστοιχο XML. Η συχνότητα των αποκομμάτων που αυτό δεν ισχύει, είναι πολύ μικρή, και είναι της τάξης του 1‰.

6. Μια ακόμα προδιαγραφή ήταν να επεξεργάζονται τα αποκόμματα από την εφαρμογή κατά το δυνατόν με την σειρά με την οποία παράγονται από το σύστημα. Το όνομα του PDF και XML αρχείου το οποίο είναι ένας αύξον αριθμός, δεν σημαίνει απαραίτητα ότι αυτή είναι και η σειρά με την οποία τα παρήγαγε το σύστημα. Έτσι, η καλύτερη λύση ήταν η ταξινόμηση των αρχείων με βάση την ημερομηνία δημιουργίας.

5.4 Επίλογος

Έχοντας καθορίσει και τις λειτουργικές προδιαγραφές πολύ εύκολα μπορούμε να αναλύσουμε και να ρίξουμε μια ματιά στον κώδικα και να δούμε πώς όλα αυτά πραγματοποιήθηκαν προγραμματιστικά.

Κεφάλαιο 6: Διασύνδεση των δύο υποσυστημάτων – υλοποίηση εφαρμογής

6.1 Εισαγωγή

Σε αυτό το κεφάλαιο αρχικά θα αναλύσουμε την λογική της εφαρμογής και τον σκοπό της κάθε κλάσης. Στη συνέχεια θα εξεταστούν κάποια σημεία του κώδικα λεπτομερώς. Μέσα από αυτή τη διαδικασία θα δοθεί η ευκαιρία να επεξηγηθούν και κάποιες κλάσεις της Java, όπως η Thread, FilenameFilter, Logging, Concurrent κα.

6.2 Αρχείο παραμετροποίησης, αρχείο XML αποκόμματος και SAX API

Καταρχάς θα δούμε το XML αρχείο από όπου η εφαρμογή αντλεί τις παραμέτρους για την λειτουργία της. Το αρχείο αυτό ονομάζεται Configuration.xml. Στην αρχή αυτού του αρχείου ορίζεται το doctype του. Το doctype - Document Type Definition (DTD) ορίζει την δομή του XML αρχείου. Προσδιορίζει δηλαδή ποιά είναι τα elements και attributes που μπορεί να εμφανίζονται, τον τύπο τους, το αν είναι προαιρετικά ή όχι, τις default τιμές τους κλπ. Ο ορισμός αυτός των δεδομένων καθιστά εύκολη την χρήση του XML ακόμα και από τρίτους. Μπορεί δηλαδή κάποιος εύκολα να καταλάβει ποιες είναι οι μεταβλητές που μπορούν να παραμετροποιηθούν.

Επίσης, με βάση το doctype γίνεται το validation του XML αρχείου, δηλαδή γίνεται αυτόματα έλεγχος για το κατά πόσο τα δεδομένα στο XML αρχείο ακολουθούν τους κανόνες του doctype. Αυτός είναι ο δεύτερος λόγος που εξυπηρετεί το να αποθηκεύονται οι παράμετροι της εφαρμογής σε XML αρχείο. Επειδή όλοι οι βασικοί έλεγχοι γίνονται κατά το validation του XML, δεν υπάρχει η ανάγκη να γίνονται όλοι αυτοί οι έλεγχοι μέσα από τον κώδικα της εφαρμογής.

Το API το οποίο θα χρησιμοποιήσουμε για το χειρισμό του αρχείου παραμετροποίησης είναι το SAX (org.xml.sax). Το SAX είναι ένα API το οποίο

διαχειρίζεται XML αρχεία. Εναλλακτικά θα μπορούσαμε να χρησιμοποιήσουμε και κάποιο άλλο API όπως τα DOM, JAXP, JDOM, dom4j κα.

Ο parser του SAX διαβάζει όλο το xml αρχείο από την αρχή μέχρι το τέλος. Κατά την διάρκεια αυτής της διαδικασίας εντοπίζει όλα τα σημεία όπου αρχίζει ή τελειώνει ένα tag, ένα κείμενο, ένα σχόλιο κ.α. μέσα στο xml. Κάθε φορά που εντοπίζει ένα τέτοιο σημείο καλεί αυτόματα την αντίστοιχη μέθοδο (startElement, endElement κα) της κλάσης η οποία υλοποιεί το interface org.xml.sax.ContentHandler. Σε αυτές τις μεθόδους, ο προγραμματιστής ορίζει το πώς θα συμπεριφερθεί ο parser με τα δεδομένα του xml αρχείου.

Επίσης θα χρησιμοποιήσουμε το SAX API για να διαβάσουμε τα δεδομένα στα XML αρχεία που παράγονται από το παραγωγικό υποσύστημα πέρα από το αρχείο παραμετροποίησης.

6.2.1 Configuration.xml

Στο παράρτημα Β μπορείτε να δείτε το Configuration.xml αρχείο το οποίο περιέχει το doctype ορισμένο στην αρχή του εγγράφου. Στη συνέχεια του παραρτήματος ακολουθεί περιγραφή του κάθε tag.

Από το doctype αξίζει να σημειώσουμε τα ακόλουθα:

- Το πρωτεύον element «Configuration» πρέπει να περιέχει απαραίτητα το κάθε υπό element μόνο μία φορά.
- Σε κάποια attlist όπως τα «ifFilesMoreThan», «maxSizeInMB», «numberOfLogFiles» κα, ορίζεται η default τιμή τους.
- Μπορούν να υπάρχουν ένα ή περισσότερα «email» elements όπου η συμπλήρωση του υπό element «to» είναι προαιρετική, και θα πρέπει να έχει δηλωθεί ή το «body» element (μόνο μία φορά) ή το «body_sample_file» element (μόνο μία φορά) ή κανένα από τα δύο.

Όπως μπορείτε να διαπιστώσετε, στο αρχείο παραμετροποίησης υπάρχουν κάποια tag τα οποία εμφανίζονται περισσότερες των μία φορές (πχ <email>,<log>). Η σειρά με την οποία εμφανίζονται είναι σημαντική για την εφαρμογή. Για παράδειγμα, το πρώτο <email> tag αφορά το email το οποίο είναι να σταλεί από την εφαρμογή αν κάτι δεν πάει καλά με την μετατροπή του Pdf σε Djvu. Το δεύτερο αφορά τα στοιχεία για την αποστολή email στον

πελάτη κατά την έναρξη συνδρομής του. Το τρίτο την αδυναμία εισαγωγής ενός αποκόμματος στην βάση, το τέταρτο την αδυναμία δημιουργίας κάποιου νέου λογαριασμού στη βάση, και το τελευταίο, κάποιο γενικότερο λάθος (όχι σχετικό με την βάση) για το οποίο δεν μπόρεσε να εισαχθεί κάποιο απόκομμα. Η σειρά αυτή πρέπει να τηρείται.

Αντίστοιχα για τα log αρχεία όπου το πρώτο και το δεύτερο <log> tag ορίζουν τα δύο αρχεία στα οποία θα καταγράφονται τα logs ενώ το τρίτο log αρχείο είναι αποκλειστικά για την διαδικασία του συγχρονισμού.

6.2.2 ConfigurationLocation.txt

Αυτό το αρχείο κειμένου δεν είναι τίποτα άλλο παρά ένα αρχείο στο οποίο θα πρέπει να αναγράφεται η διεύθυνση στην οποία βρίσκεται το Configuration.xml αρχείο. Η εφαρμογή δηλαδή θα διαβάζει αυτό το αρχείο, και στη διεύθυνση που θα περιέχεται σε αυτό, θα αναζητήσει το αρχείο παραμετροποίησης.

Ο λόγος που επιλέχθηκε αυτή η υλοποίηση είναι γιατί δίνει την δυνατότητα να αποθηκεύεται το αρχείο παραμετροποίησης σε διεύθυνση διαφορετική από την διεύθυνση εγκατάστασης της εφαρμογής. Αντίθετα, η εφαρμογή θα αναζητά το αρχείο ConfigurationLocation.txt σε συγκεκριμένη διεύθυνση ορισμένη στον κώδικα.

6.2.3 ConfigurationContentHandler.java

Η κλάση αυτή ορίζει το πώς ο parser θα διαβάσει τα περιεχόμενα του αρχείου παραμετροποίησης. Κατά την διαδικασία του parsing τα δεδομένα που διαβάζονται αποθηκεύονται στις τοπικές μεταβλητές της κλάσης.

Η κλάση ConfigurationContentHandler.java υλοποιεί το interface org.xml.sax.ContentHandler. Ένας ContentHandler πρακτικά ορίζει το πώς ο parser θα χειριστεί τα περιεχόμενα του αρχείου. Η υλοποίηση απαιτεί να γίνουν implement οι ακόλουθες μέθοδοι:

- public void characters(char[] text, int start, int length)
- public void startElement(String namespaceURI, String localName, String qualifiedName, Attributes atts)
- public void endElement(String namespaceURI, String localName, String qualifiedName)

- public void setDocumentLocator(Locator locator)
- public void startDocument()
- public void endDocument()
- public void startPrefixMapping(String prefix, String uri)
- public void endPrefixMapping(String prefix)
- public void ignorableWhitespace(char[] text, int start, int length) throws SAXException
- public void processingInstruction(String target, String data)
- public void skippedEntity(String name)

Στην περίπτωσή μας, χρειάζονται μόνο οι τρεις πρώτες μέθοδοι, οι οποίες έχουν υλοποιηθεί, ενώ οι υπόλοιπες απλά έχουν οριστεί και το σώμα τους παραμένει κενό.

Η startElement(String namespaceURI, String localName, String qualifiedName, Attributes atts) καλείτε από τον Parser αυτόματα κάθε φορά με το που ξεκινάει ένα tag (πχ <general_input>). Οι εργασίες που συνήθως απαιτούνται να πραγματοποιηθούν με το που ξεκινάει ένα tag ενός element είναι το διάβασμα των attributes και η αρχικοποίηση μεταβλητών. Στην παράμετρο qualifiedName έχει περαστεί το όνομα του tag (element) και στο atts τα attributes αυτού. Στο αρχείο ConfigurationContentHandler.java περιέχεται το απόσπασμα κώδικα που απεικονίζεται στον *Κώδικας 11: Μέθοδος startElement*.

```
public void startElement(String namespaceURI, String localName,
 String qualifiedName, Attributes atts) {
      if(qualifiedName.equals("general_input")){
          interval = Integer.parseInt(atts.getValue("interval")+"000");//μετατροπή σε second
      }
      if(qualifiedName.equals("redirectionPath")){
          ifFilesMoreThanLimit = Integer.parseInt(atts.getValue("ifFilesMoreThan"));
      }
      if(qualifiedName.equals("email_config")){
          i=0;
           emailCont = new EmailContainer();

      }
      if(qualifiedName.equals("log")){
          i=0;
          logFiles_number++;
          logAttr_maxSizeInMB = atts.getValue("maxSizeInMB");
          logAttr_numberOfLogFiles = atts.getValue("numberOfLogFiles");
          logAttr_append = atts.getValue("append");
      }
}
```

Κώδικας 11: *Μέθοδος startElement*

Κάθε φορά που καλείται η startElement (δηλαδή κάθε φορά που ξεκινάει το οποιοδήποτε Element) πρέπει να εντοπιστεί μέσα από μια σειρά εντολών ελέγχου ποιο είναι το tag το οποίο κάλεσε την μέθοδο και να προβούμε με βάση αυτό στις κατάλληλες ενέργειες. Συγκεκριμένα, για το tag «general_input» διαβάζουμε την τιμή που έχει το attribute «interval» του element ενώ για το tag «email_config» δημιουργούμε έναν πίνακα ο οποίος θα υποδεχθεί αντικείμενα τύπου EmailContainer (βλ. κλάση EmailContainer.java), δηλαδή τις επιμέρους παραμετροποιήσεις για τα email.

Ο parser για το κάθε σημείο στο XML όπου αρχίζει το περιεχόμενο ενός tag καλεί την μέθοδο characters(char[] text, int start, int length). Ο πίνακας text περιέχει κάθε φορά όλους τους χαρακτήρες του xml, ο αριθμός start δείχνει σε ποιο σημείο του xml βρίσκεται την τρέχουσα στιγμή (δηλαδή τον πρώτο χαρακτήρα των δεδομένων του τρέχοντος tag) και το length το μέγεθός του δεδομένου σε χαρακτήρες. Αποθηκεύοντας τους χαρακτήρες του πίνακα από το σημείο start και μέχρι το σημείο start+length, αποθηκεύουμε πρακτικά το περιεχόμενο του tag όπως δείχνει ο ακόλουθος κώδικας.

```
public void characters(char[] text, int start, int length)
 throws SAXException {
    tmp=String.valueOf(text,start,length);
}
```

Κώδικας 12: Μέθοδος characters

Η μέθοδος endElement(String namespaceURI, String localName, String qualifiedName) καλείται κάθε φορά που τελειώνει ένα tag (για παράδειγμα </general_input>). Όμοια, μέσα από σειρά εντολών ελέγχου, πρέπει να εντοπιστεί ποιο είναι το tag το οποίο κάλεσε την μέθοδο. Όταν ο parser φτάσει να διαβάσει το τέλος ενός tag σημαίνει πως έχει διαβάσει και τα δεδομένα τα οποία περικλείει το tag και έχει καλέσει την μέθοδο characters. Άρα, πρέπει να αποθηκευτούν τα δεδομένα που διάβασε η characters μέθοδος στις τοπικές μεταβλητές.

```
public void endElement(String namespaceURI, String localName,
String qualifiedName) {
    if(qualifiedName.equals("general_input")){
        general_input = tmp;
    }
    if(qualifiedName.equals("redirectionPath")){
        redirectionPath = tmp;
    }
    if(qualifiedName.equals("pdf_output")){
        pdf_output = tmp;
    }
```

Κώδικας 13: *Μέθοδος startElement()*

6.2.4 ReadXMLConf.java

Σε αυτή την κλάση πραγματοποιείται το parse του αρχείου παραμετροποίησης με βάση τον ContentHandler.java και την χρήση της κλάσης org.xml.sax.XMLReader όπως φαίνεται και στον *Κώδικας 14: Parsing το XML αρχείο παραμετροποίησης*:

```
public class ReadXMLConf {
    private String source = "file:///";
    private ConfigurationContentHandler contentHandl = null;
    public ReadXMLConf(String src){
        source += src;
        try{
           XMLReader parser = XMLReaderFactory.createXMLReader();
           contentHandl = new ConfigurationContentHandler();

           parser.setContentHandler(contentHandl);
           parser.parse(new InputSource(source));
```

Κώδικας 14: *Parsing το XML αρχείο παραμετροποίησης*

6.2.5 DistributionXmlObj.java

Σε αντικείμενο αυτής της κλάσης αποθηκεύονται τα δεδομένα των αποκομμάτων τα οποία είναι σε XML μορφή, αφού πρώτα τα XML γίνουν Parse από τον SAX parser. Ενδιαφέρον παρουσιάζει η μέθοδος setTenderFields() η οποία ελέγχει τον τύπο του αποκόμματος και καθορίζει αν αυτό είναι άρθρο ή δημοπρασία.

```
public void setTenderFields(){
    if(!articleType.equals("2")){//άρθρο
        tenderExpireDate = "null";
        tenderAmount = "null";
        foreas = "null";
    }
    else{//δημοπρασία
        setTenderExpireDate() ;
        setTenderAmount() ;
        headline = "null";
    }
}
```

Κώδικας 15: *Μέθοδος setTenderFields*

6.2.6 PrepareFile.java

Για να γίνει parse ένα xml αρχείο με επιτυχία απαιτείται στα δεδομένα να μην υπάρχουν ειδικοί χαρακτήρες. Ειδικοί χαρακτήρες για τα XML αρχεία είναι

οι: &, <, >, " και '. Για να γίνει Parse το αρχείο θα πρέπει οι χαρακτήρες αυτοί να αντικατασταθούν με τα αντίστοιχα ισοδύναμά τους που είναι τα: &, <, >, ", '. Αυτός είναι ο λόγος για τον οποίο δημιουργήθηκε η κλάση PrepareFile.java. Ο τίτλος και το κείμενο OCR ενός άρθρου είναι τα μόνα σημεία στο XML αρχείο στα οποία είναι πιθανό να βρίσκονται ειδικοί χαρακτήρες. Έτσι, πριν προχωρήσουμε στο parse, το αρχείο πρέπει πρώτα να ελεγχθεί μέσω της κλάσης PrepareFile.java και εφ' όσον υπάρχουν ειδικοί χαρακτήρες να αντικατασταθούν με τα ισοδύναμά τους. Μετά την ολοκλήρωση αυτής της διαδικασίας, δημιουργείται ένα xmlFilename.xml_tmp αρχείο όπου αποθηκεύεται το αρχικό xml διορθωμένο. Στην συνέχεια δημιουργείται το xmlFilename.xml_tmp2 το οποίο είναι ίδιο με το xmlFilename.xml_tmp αλλά είναι σε UTF-16 encoding. Έτσι, το αρχικό (xmlFilename.xml) με το τελικό αρχείο xml (xmlFilename.xml_tmp2) είναι στο ίδιο encoding, αλλά στο τελικό xml, έχουν αντικατασταθεί οι ειδικοί χαρακτήρες. Στην πρώτη γραμμή του XML αρχείου (<?xml version="1.0" encoding="UTF-16" ?>) ορίζεται το encoding του. Η κλάση επιστρέφει την διεύθυνση του xmlFilenamelxml_tmp2 στην έξοδό του.

6.2.7 DistributionContentHandler.java

Η χρήση της DistributionContentHandler.java είναι ίδια με την χρήση της ConfigurationContentHandler.java αλλά για το XML αρχείο του αποκόμματος. Δηλαδή η DistributionCon

tentHandler.java είναι ένας org.xml.sax.ContentHandler ο οποίος καθορίζει το πώς θα διαβαστεί το αρχείο xml του παραγωγικού υποσυστήματος. Τέλος, αποθηκεύει τα δεδομένα του xml αρχείου σε αντικείμενο τύπου DistributionXmlObj.java.

Στον κώδικα υπάρχει μόνο μία διαφοροποίηση και αφορά την επιπρόσθετη μέθοδο checkString(string str) η οποία τοποθετεί εκ νέου τους ειδικούς χαρακτήρες στις μεταβλητές του τίτλου και του OCR. Το & για παράδειγμα αντικαθίσταται από το & πριν την εκχώρηση του δεδομένου σε αντικείμενο τύπου DistributionObj.java.

6.2.8 ReadDistributionXML.java

Αντίστοιχη με τη ReadXMLConf.java είναι και η κλάση ReadDistributionXML.java η οποία αφού χρησιμοποιήσει το xml αρχείο του παραγωγικού υποσυστήματος από την PrepareFile.java για την αντικατάσταση των ειδικών χαρακτήρων, κάνει parse το XML και διαγράφει το προσωρινό xmlFilename.xml_tmp2 αρχείο.

6.3 Λίστα περιεχομένων των αρχείων προς επεξεργασία.

Όπως έχουμε αναφέρει η παραγωγική διαδικασία της αποδελτίωσης έχει ως αποτέλεσμα την παραγωγή δύο αρχείων. Ένα PDF και ένα XML. Το PDF αποτελεί το απόκομμα του πελάτη και το XML πληροφορίες για το απόκομμα. Τα δύο αυτά αρχεία καταλήγουν σε ένα φάκελο, σε κάποιον server στο δίκτυο στον οποίο έχει πρόσβαση η εφαρμογή. Μέσα σε ένα δευτερόλεπτο μπορούν να καταφτάσουν πολλές τέτοιες δυάδες αρχείων, ειδικά σε ώρες αιχμής. Τα αρχεία αυτά τα οποία πρέπει να επεξεργαστούν από την εφαρμογή μας αποτελούν το input της. Είναι σημαντικό τα αρχεία αυτά να επεξεργάζονται με συγκεκριμένη σειρά. Αν για παράδειγμα στον φάκελο έχουν συσσωρευτεί 100 δυάδες αρχείων, θέλουμε αυτά που δημιουργήθηκαν-αποδελτιώθηκαν πρώτα, να επεξεργαστούν και πρώτα, για να είναι προσβάσιμα από τους πελάτες. Ο λόγος είναι ότι υπάρχουν έντυπα/εφημερίδες μεγαλύτερης σπουδαιότητας από κάποια άλλα. Η αποδελτίωση γίνεται με βάση την προτεραιότητα αυτή. Δηλαδή πρώτα αποδελτιώνονται τα σημαντικότερα έντυπα και άρα κόβονται πρώτα τα αποκόμματα από αυτά τα έντυπα. Έτσι, είναι σημαντικό αυτά τα αποκόμματα να είναι διαθέσιμα πρώτα στο portal του Web έναντι των υπολοίπων.

Το όνομα των αρχείων είναι ένας αριθμός (id), έστω 123456.PDF και 123456.XML . Αν ένα αρχείο έχει μεγαλύτερο id από ένα άλλο, δεν σημαίνει ότι αποδελτιώθηκε απαραίτητα αργότερα από αυτό που έχει μικρότερο id. Έστω το αρχείο Α που αποδελτιώνεται την χρονική στιγμή t και είναι ένα τετρασέλιδο απόκομμα, και έστω το αρχείο Β που αποδελτιώνεται τη χρονική t+x (όπου x,t>0) και είναι μονοσέλιδο απόκομμα. Από τη στιγμή που θα κοπεί

το απόκομμα επεξεργάζεται από ένα σύνολο εφαρμογών του παραγωγικού συστήματος για την προσθήκη του OCR, την υπογράμμιση της λέξεις κλειδιού κα. Επειδή το απόκομμα Α είναι μεγαλύτερο σε μέγεθος από το Β θα χρειαστεί περισσότερο χρόνο στην επεξεργασία του. Έτσι, το Β θα φτάσει πρώτο για επεξεργασία από την τελική εφαρμογή η οποία θα δημιουργήσει και θα στείλει το Pdf και XML αρχείο στο φάκελο και θα έχει μικρότερο id από το Α το οποίο αποδελτιώθηκε πρώτο.

Διαπιστώθηκε πειραματικά ότι η ταξινόμηση των αρχείων με βάση την ημερομηνία τροποποίησης είναι προσεγγιστικά πιο κοντά στην πραγματικότητα και με αυτή τη σειρά θα πρέπει να επεξεργάζονται από την εφαρμογή.

Άρα, το πρώτο πράγμα που θα πρέπει να κάνει η εφαρμογή είναι να πάρει τα περιεχόμενα του φακέλου ταξινομημένα με την ημερομηνία τροποποίησης. Αυτό που θα πρέπει να φροντίσουμε είναι να δημιουργήσουμε ένα φίλτρο έτσι ώστε να πάρει μόνο τα ids των PDF αρχείων. Αν δεν χρησιμοποιούσαμε το φίλτρο θα είχε ως αποτέλεσμα να έχουμε το κάθε id δύο φορές. Επίσης αν δεν χρησιμοποιούσαμε το φίλτρο θα προκαλούνταν πρόβλημα αν για κάποιο λόγω κάποιο άσχετο αρχείο ή φάκελος δημιουργούνταν στο φάκελο που βρίσκονται τα αποκόμματα. Εναλλακτικά θα μπορούσαμε να χρησιμοποιήσουμε φίλτρο XML αρχείων, όμως αυτό κρύβει ένα μεγάλο πρόβλημα. Τα XML αρχεία έχουν πολύ μικρότερο μέγεθος από τα PDF. Έτσι, η μεταφορά τους τελειώνει πολύ πιο γρήγορα. Τα αρχεία θα παίρνονται και θα επεξεργάζονται ανά δυάδες. Αν δεν έχει ολοκληρωθεί η μεταφορά του PDF και η εφαρμογή μας βρίσκοντας το XML προσπαθήσει να επεξεργαστεί το αρχείο αυτό, θα αντιμετόπιζε πρόβλημα γιατί δεν θα μπορούσε να βρεί το απόκομμα. Αυτό δυστυχώς συμβαίνει αρκετές φορές και κυρίως όταν υπάρχουν διαθέσιμα thread (δηλαδή όταν δεν υπάρχει φόρτος) τα οποία θα πάρουν άμεσα το κάθε επόμενο αρχείο προς επεξεργασία.

6.3.1 LastModifiedFileComparator.java

Η κλάση αυτή υλοποιεί τη σύγκριση μεταξύ δύο αντικειμένων τύπου Object μέσω της μεθόδου compare(Object obj1, Object obj2) σχετικά με το αν το

πρώτο είναι μεταγενέστερο του δεύτερου. Σε αυτή τη περίπτωση επιστρέφει: 1. Αν είναι προγενέστερο επιστρέφει –1 και αν η ημερομηνία τροποποίησης μεταξύ των δύο αντικειμένων είναι ίδια, επιστρέφει 0.

6.3.2 PdfExtensionFilter.java.

Εδώ ορίζεται ένα PDF φίλτρο το οποίο θα επιστρέφει μόνο ονόματα αρχείων που έχουν κατάληξη «.PDF» ή «.pdf».

6.3.3 CheckAndGetSortedList.java.

Η κλάση CheckAndGetSortedList.java ελέγχει για PDF αρχεία μέσα σε μία διεύθυνση φακέλου και στην συνέχεια τα ταξινομεί με βάση την ημερομηνία τροποποίησης. Επιστρέφει έναν πίνακα ακεραίων ο οποίος περιέχει τα ονόματα των αρχείων χωρίς την κατάληξη αυτών. Αν δεν υπάρχουν PDF αρχεία στην διεύθυνση του φακέλου επιστρέφει null.

6.4 Logging

Ένα σημαντικό κομμάτι κάθε προγράμματος είναι η καταγραφή των σφαλμάτων ή και των διεργασιών που διαπεραιώθηκαν με επιτυχία. Η καταγραφή αυτή πραγματοποιείται κατά κύριο λόγο σε κάποιο αρχείο. Η εφαρμογή καταγράφει κάποιες βασικές πληροφορίες για τα αρχεία τα οποία επεξεργάζεται όπως το αν η μετατροπή του PDF αρχείου σε DjVu έγινε επιτυχώς, το αν ενημερώθηκε η βάση κα. Έτσι, το log αρχείο πολύ γρήγορα μπορεί να αυξηθεί δραματικά σε μέγεθος και κάποια στιγμή, όσο αυξάνετε κάθε λεπτό σε μέγεθος, να γεμίσει όλο το διαθέσιμο χώρο στο δίσκο. Έτσι, είναι βασικό τα log αρχεία να μην υπερβαίνουν ορισμένο μέγεθος. Αν θέλουμε μεγάλο log αρχείο θα πρέπει να μπορούμε να μοιράζουμε τις εγγραφές μεταξύ περισσότερων του ενός αρχείων. Αν δηλαδή χρειάζεται να έχουμε αρχείο log εγγραφών ενός μήνα και αυτό αντιστοιχεί σε αρχείο κειμένου μεγέθους 10MB, είναι καλύτερο το μέγεθος αυτό να μοιραστεί σε 5 αρχεία των

2MB. Όταν και το τελευταίο αρχείο γεμίσει, οι επόμενες εγγραφές εισάγονται στο πρώτο αρχείο διαγράφοντας τις παλαιότερες εγγραφές σε αυτό.

Επιπρόσθετα, για την καταχώρηση των εγγραφών στα log αρχεία χρησιμοποιήθηκε ανακατεύθυνση των System.out και System.err, έτσι ώστε κάθε φορά που χρησιμοποιείται η System.out.println() και κάθε φορά που χτυπάει κάποιο error, αυτό να εγγράφεται απευθείας στα log αρχεία. Η υλοποίηση του Logging έγινε με χρήση του πακέτου java.util.logging.

Η εφαρμογή χρησιμοποιεί τρία αρχεία για την καταγραφή των logs όπως φανερώνει και το αρχείο παραμετροποίησης. Το πρώτο και το δεύτερο εξυπηρετούν την ίδια ανάγκη, δηλαδή την καταγραφή όλων των μηνυμάτων που παράγονται κατά την επεξεργασία των αποκομμάτων και το τρίτο κρατάει τα logs της διαδικασίας του συγχρονισμού.

6.4.1. StdOutErrLevel.java

Η κάθε log εγγραφή πραγματοποιείται από ένα Logger αντικείμενο και χαρακτηρίζεται από ένα logging Level. Το logging Level αφορά την σημαντικότητα της log εγγραφής και μπορεί να είναι FINEST, FINER, FINE, CONFIG, INFO, WARNING και SEVERE (όπου το πρώτο υποδηλώνει την μικρότερη σημαντικότητα ενώ το τελευταίο τη μεγαλύτερη). Η κλάση StdOutErrLevel.java δημιουργεί δύο νέα Level, με όνομα STDOUT και STDERR, και τα οποία θα χρησιμοποιηθούν έναντι των προκαθορισμένων Levels. Ο λόγος που εφαρμόζεται συνήθως αυτή η πρακτική είναι για να ξεχωρίζουν εύκολα τα logs αυτά από τα logs των υπόλοιπων Loggers..

6.4.2 LoggingOutputStream.java

Η κλάση αυτή δημιουργεί ένα Stream εξόδου με δοσμένο Logger και level. Έτσι, δημιουργώντας αντικείμενο τύπου LoggingOutputStream και δίνοντας τους ως παράμετρο το Logger αντικείμενο και τα “STDERR” ή “STDOUT” που έχουμε δημιουργήσει, έχει ως αποτέλεσμα ένα output stream που γράφει log εγγραφές με τα ορισμένα από εμάς level. Αξίζει να σημειώσουμε ότι για να αποφύγουμε περιττές «κενές» εγγραφές, δεν εγγράφονται εγγραφές στο log

αν κληθεί η μέθοδος flush() δύο συνεχόμενες φορές ή όταν είναι να εγγραφεί κενή γραμμή.

6.4.3 LoggingInitialization.java

Οι Loggers έχουν ιεραρχική δομή και στην κορυφή είναι ο root logger (ο οποίος ονομάζεται ""). Οι υπόλοιποι Loggers κληρονομούν από αυτόν κάποια βασικά στοιχεία όπως το Logging Level και το Handler. Ο Logger μεταφέρει τις εγγραφές στον Handler. Ο Handler αποφασίζει πού θα καταγραφούν οι εγγραφές. Ο Handler που θα χρησιμοποιήσουμε είναι ο FileHandler ο οποίος καταγράφει τις εγγραφές σε ένα αρχείο ή σε μια σειρά από κυλιόμενα log αρχεία. Ορίζοντας τον root Logger να χρησιμοποιεί τον FileHandler πρακτικά ορίζονται όλοι οι Loggers να χρησιμοποιούν το/τα αρχεία που ορίζει το FileHandler για την καταγραφή των log εγγραφών, μέσω της κληρονομικότητας.

```
Handler fileHandler = new FileHandler(logFile, sizeInMB * 1000000, NumberOfFiles, toAppend);
fileHandler.setFormatter(new SimpleFormatter());
Logger.getLogger("").addHandler(fileHandler);
```

Κώδικας 16: *Δημιουργία FileHandler*

Τα System.out και System.err είναι PrintStream αντικείμενα. Αυτό που χρειάζεται είναι να ανακατευθύνουμε την έξοδό τους στο δικό μας PrintStream που είναι το LoggingOutputStream που αναφέραμε παραπάνω.

```
Logger logout = Logger.getLogger("stdout");
Logger loger = Logger.getLogger("stderr");
LoggingOutputStream losout = new LoggingOutputStream(logout, StdOutErrLevel.STDOUT);
LoggingOutputStream loserr = new LoggingOutputStream(loger, StdOutErrLevel.STDERR);
System.setOut(new PrintStream(losout, true));
System.setErr(new PrintStream(loserr, true));
```

Κώδικας 17: *Ανακατεύθυνση stream εξόδου και λάθους*

Έτσι, κάθε φορά που εκτελείται μία System.out.println() εντολή, αυτή θα καταχωρείται στο log αρχείο με Level STDOUT όπως και όλα τα μηνύματα λάθους με Level STDERR.

6.5 Email

Όπως έχουμε αναφέρει η εφαρμογή θα πρέπει να στέλνει email στις ακόλουθες περιπτώσεις:

1. Όταν ξεκινάει ένας πελάτης, πρέπει να στέλνεται ένα ενημερωτικό email όπου θα δίνονται οδηγίες για το πώς ο πελάτης μπορεί να αποκτήσει πρόσβαση στην αποδελτίωσή του μέσω του site, παραθέτοντάς του ταυτόχρονα το username και password , το οποίο πρέπει να εισάγει στο site προκειμένου να συνδεθεί. Το email αυτό επίσης πρέπει να κοινοποιείται σε μια διεύθυνση ηλεκτρονικού ταχυδρομείου της εταιρείας αποδελτίωσης προς ενημέρωση και επιβεβαίωση της έναρξης.
2. Στην περίπτωση που κάτι δεν πήγε καλά κατά την εισαγωγή αποκόμματος στο υποσύστημα δημοσίευσης ή κατά τη δημιουργία του PDF αρχείου σε DjVu ή κατά την διαδικασία του συγχρονισμού.

6.5.1 EmailFields.java

Η κλάση αυτή αναπαριστά τα στοιχεία/δεδομένα που απαρτίζουν ένα email, όπως το περιεχόμενο, τον αποστολέα, το θέμα κλπ. Έτσι, ένα αντικείμενο αυτής της κλάσης πρακτικά αντιπροσωπεύει ένα email.

6.5.2 EmailContainer.java

Σε αντικείμενο αυτής της κλάσης η ConfigurationContentHandler.java αποθηκεύει όλα τα email τα οποία έχουν οριστεί από το Configuration.xml αρχείο παραμετροποίησης. Με το που ο Parser θα βρεί το “email_config” tag θα δημιουργήσει ένα αντικείμενο EmailContainer. Στη συνέχεια, διαβάζοντας τα tags των emails (host, from, to κα) καταχωρεί τις πληροφορίες αυτές σε ένα EmailFields αντικείμενο. Κάθε φορά που βρίσκει το σημείο όπου κλείνει το tag «email» αποθηκεύει το EmailFields αντικείμενο στον EmailContainer.

6.5.3 Mail.java

Η κλάση Mail.java διαχειρίζεται την αποστολή των email μέσο του πακέτου javax.mail.

6.5.4 NewClientEmail.java

Το email ενημέρωσης έναρξης πελάτη διαφοροποιείται από τα υπόλοιπα σε μερικά σημεία. Καταρχάς πέρα από τον πελάτη πρέπει να σταλεί σε ακόμα μία διεύθυνση, η οποία θα ανήκει στην εταιρεία αποδελτίωσης. Η διεύθυνση του πελάτη φυσικά δεν μπορεί να περιέχεται στο αρχείο παραμετροποίησης. Έτσι στο αρχείο παραμετροποίησης ορίζεται μόνο η διεύθυνση της εταιρείας αποδελτίωσης. Το email του πελάτη θα δίνεται παραμετρικά στον δομητή της κλάσης NewClientEmail.java. Επιπλέον, στο αρχείο παραμετροποίησης ορίζεται το sample αρχείο το οποίο θα χρησιμοποιηθεί για το body του email και το αρχείο αυτό θα μπορούσε να είναι το newClient_emailSample.txt. Σε αυτό, υπάρχει μία γραμμή η οποία αποτελείται μόνο από τον χαρακτήρα '@'. Αυτή είναι η θέση όπου η εφαρμογή πρέπει να συμπληρώσει το username και το password του πελάτη. Αυτή η διαδικασία γίνεται από την μέθοδο edit_newClient_Body η οποία καλείται από τον δομητή και παίρνει ως παράμετρο το δείγμα αρχείου και αντικαθιστά το χαρακτήρα '@' με τυχαίους αριθμούς που δημιουργεί η εφαρμογή και επιπρόσθετο συμπληρωματικό κείμενο. Τέλος με μια απλή μέθοδο όπως η ακόλουθη (βλ *Κώδικας 18: Αποστολή Email*) μπορούμε να στείλουμε το email στον νέο πελάτη.

```
private void sendEmail2newClient(String customer_email){
    EmailContainer ec = contHand.getEmailCont();
    EmailFields ef = ec.getEmail(1);
    NewClientEmail nce = new NewClientEmail(ef,customer_email);
    Mail m = new Mail(nce);
    m.sendMail();
}
```

Κώδικας 18: *Αποστολή Email*

6.6 Αλληλεπίδραση με τις βάσεις δεδομένων.

Στη συνέχεια θα περιγράψουμε τις κλάσεις στις οποίες έχει υλοποιηθεί η αλληλεπίδραση με τις βάσεις δεδομένων. Συγκεκριμένα έχουμε δύο κλάσεις για την άντληση δεδομένων από την Oracle και δύο για την άντληση/προσθήκη/τροποποίηση δεδομένων στην MSSQL. Η ConnectAtOracleDB.java είναι βοηθητική κλάση η οποία πραγματοποιεί την σύνδεση με την Oracle, εκτελεί βασικά query τα οποία της δίνονται παραμετρικά και τέλος αποθηκεύει τα αποτελέσματα. Όμοια, η ConnectAtSqlDB.java υλοποιεί την σύνδεση με την MSSQL, εκτελεί κάποια βασικά query και αποθηκεύει τα δεδομένα. Οι κλάσεις OracleDatabaseQueries.java και SQLDatabaseQueries.java υλοποιούν τα πιο σύνθετα ερωτήματα προς την Oracle και την MSSQL αντίστοιχα.

6.6.1 Company.java

Η βοηθητική αυτή κλάση περιγράφει έναν πελάτη της εταιρείας αποδελτίωσης. Περιέχει δηλαδή τα στοιχεία που τον απαρτίζουν, όπως το όνομα της εταιρείας, τον κωδικό, την τοποθεσία, τις λέξεις κλειδιά για τις οποίες ενδιαφέρεται κα.

6.6.2 ConnectAtSqlDB.java

Η ConnectAtSqlDB.java είναι μια βοηθητική κλάση η οποία μέσω της μεθόδου Connect() συνδέεται στην MSSQL και εκτελεί query μέσω της Excecute(String SQL). Εδώ αξίζει να σημειώσουμε ότι η Excecute έχει την δυνατότητα να εκτελέσει και εντολές Insert και Update πέρα από εντολές Select. Στην συνέχεια, τα αποτελέσματα αποθηκεύονται στις μεταβλητές μελους.

6.6.3 SQLDatabaseQueries.java

Σε αυτή την κλάση βρίσκονται συγκεντρωμένες όλες οι μέθοδοι οι οποίες αντλούν δεδομένα από την βάση της MSSQL. Ακολουθεί μία σύντομη περιγραφή της κάθε μεθόδου.

- getAllCustomers(): Επιστρέφει έναν πίνακα με όλους τους πελάτες της βάσης (ενεργοί και μη ενεργοί).
- getAllCustomersStr(): Επιστρέφει ένα String το οποίο επιτρέπει την χρήση του String αυτού σε SQL εκφράσεις τύπου «where customer_code in (“AA1”,”BB2”,”CC3”). Με βάση το προηγούμενο παράδειγμα, η getAllCustomers() θα επιστρέψει το new String{“AA1”,”BB2”,”CC3”}, ενώ η getAllCustomersStr() θα επιστρέψει το «new String(”(/“AA1/”,/”BB2/”,/”CC3/”)”)». Ο λόγος που χρειάζεται αυτή η μέθοδος είναι επειδή θέλουμε τα αποτελέσματα αυτά να τα ενσωματώσουμε σε query το οποίο εκτελείται στην Oracle.
- createAccount(Company newClient, Boolean sendEmail): Η μέθοδος αυτή χρησιμοποιείται σε δύο περιπτώσεις, α) κατά την δημιουργία νέου πελάτη και β) κατά την εισαγωγή των νέων λέξεων κλειδιών υπάρχοντος πελάτη. Στην πρώτη περίπτωση θέλουμε να αποστέλλεται email στον νέο πελάτη για την έναρξη της συνδρομής του, ενώ στην δεύτερη περίπτωση όχι, αφού απλά θέλουμε να προστεθούν κάποιες λέξεις κλειδιά. Έτσι, όταν ισχύει η πρώτη περίπτωση η μεταβλητή sendEmail θα έχει την τιμή «true» και στην δεύτερη περίπτωση θα έχει την τιμή «false».

Πρακτικά για την βάση, η εισαγωγή ενός keyword είναι ίδια με αυτή της εισαγωγής ενός πελάτη για τον ακόλουθο λόγο: όταν εισάγεται ένας νέος πελάτης, πρέπει υποχρεωτικά να εισαχθούν και οι λέξεις κλειδιά για τις οποίες ενδιαφέρεται. Το stored procedure «sp_insertKeyword» είναι αυτό που υλοποιεί αυτή την διαδικασία. Πρέπει να τονίσουμε πως κάθε φορά που θέλουμε να εισάγουμε είτε νέο πελάτη είναι νέα keywords και στις δύο περιπτώσεις δίνονται τα ίδια στοιχεία στο stored procedure, και είναι όλες οι πληροφορίες που αφορούν τον πελάτη και όλες οι πληροφορίες που αφορούν την λέξη κλειδί. Το stored procedure εισάγει τα δεδομένα μόνο εφ’ όσον δεν υπάρχουν ήδη. Στην περίπτωση που κάποια δεδομένα έχουν τροποποιηθεί

(πχ αλλάξει η επωνυμία της εταιρείας) το stored procedure το εντοπίζει και πραγματοποιεί την αλλαγή. Έτσι, στην περίπτωση της δημιουργίας νέου πελάτη, το stored procedure βλέπει ότι ο πελάτης αυτός δεν υπάρχει και τον εισάγει. Στην συνέχεια εισάγει και τα keywords για τα οποία ενδιαφέρεται. Στην δεύτερη περίπτωση, βλέπει ότι ο πελάτης υπάρχει, και απλά εισάγει τις λέξεις κλειδιά που δεν έχουν εισαχθεί.

- getStoppedCustomers(): Επιστρέφει έναν πίνακα από String ο οποίος περιέχει τους πελάτες οι οποίοι στην βάση της MSSQL έχουν δηλωθεί ως μη ενεργοί. Ένας πελάτης θεωρείται ανενεργός για την βάση όταν στο πεδίο contract_expiration_date δεν είναι null και είναι συμπληρωμένη η ημερομηνία απενεργοποίησης του πελάτη.

- GetStoppedCustomersStr(): Η μέθοδος αυτή καλεί την getStoppedCustomers() και μετατρέπει τα αποτελέσματά της σε String το οποίο να μπορεί να χρησιμοποιηθεί σε εκφράσεις « … where customer_code IN GetStoppedCustomersString» (όπως ακριβώς κάνει η getAllCustomersStr()). Δηλαδή δημιουργεί ένα String το οποίο διαχωρίζει τους πελάτες με κόμμα και το περικλύζει σε παρενθέσεις.

- getActiveCustomers(): Επιστρέφει έναν πίνακα με τους ενεργούς πελάτες. Ένας πελάτης θεωρείται ενεργός όταν το contract_expiration_date είναι διάφορο του null.

- getActiveCustomersStr(): Επιστρέφει ένα String το οποίο διαχωρίζει τους ενεργούς πελάτες με κόμμα και τους περικλείει σε παρενθέσεις.

- formatSQLsBracket(String[] results): Η private βοηθητική αυτή μέθοδος χρησιμοποιείται από τις getAllCustomersStr(), GetStoppedCustomersStr(), getActiveCustomersStr() και υλοποιεί την μετατροπή του πίνακα σε String. Συγκεκριμένα, δημιουργεί ένα String το οποίο περιέχει όλα τα περιεχόμενα του πίνακα διαχωρισμένα με κόμμα, εισάγει μονό εισαγωγικό στην αρχή και στο τέλος του κάθε περιεχομένου και περικλείει όλο αυτό σε παρένθεση. Επιστρέφει το String το οποίο σχηματίζεται.

- activateCompany(String code): Καλεί το stored procedure «sp_activateCompany». Το stored procedure αυτό ενεργοποιεί τον πελάτη που εισάγεται παραμετρικά. Δηλαδή αλλάζει την τιμή του πεδίου contract_expiration_date σε null.

- deactivateCompany(String code): Καλεί το stored procedure «sp_deactivateCompany». Το stored procedure αυτό απενεργοποιεί τον πελάτη που εισάγεται παραμετρικά. Δηλαδή αλλάζει την τιμή του πεδίου contract_expiration_date, εισάγοντας την τρέχουσα ημερομηνία.

- insertClip(DistributionXmlObj xmlObj): Ελέγχει τον τύπο του αποκόμματος (δηλαδή αν είναι άρθρο ή δημοπρασία). Ανάλογα με τον τύπο, καλεί την αντίστοιχη μέθοδο για την εισαγωγή του αποκόμματος αφού όμως πρώτα μετατρέψει τις μεταβλητές ημερομηνίας από τύπο String σε τύπο java.sql.Timestamp. Ο τύπος timestamp της java είναι ο αντίστοιχος του datetime της MSSQL.

- insertArticle(…): Καλεί το stored procedure «sp_insertArticle» δίνοντας του όλα τα δεδομένα που χρειάζονται για την εισαγωγή του άρθρου.

- insertTender(…): Καλεί το stored procedure«sp_insertTender» δίνοντάς του όλα τα δεδομένα τα οποία χρειάζονται για την εισαγωγή της δημοπρασίας.

- sendEmail2newClient(String customer_email): Στέλνει email στον νέο πελάτη σχετικά με την ενεργοποίηση του λογαριασμού του.

- sendEmail2admin(ConfigurationContentHandler contHand, Company company): Στέλνει email στον διαχειριστή του συστήματος της εταιρείας αποδελτίωσης έτσι ώστε να επιληφθεί κάποιο πρόβλημα κατά την εισαγωγή κάποιου αποκόμματος.

6.6.4 ConnectAtOracleDB.java

Η βοηθητική αυτή κλάση διαθέτει την μέθοδο Connect() η οποία πραγματοποιεί σύνδεση με την βάση. Επιπρόσθετα, διαθέτει και την μέθοδο Execute(String sql) η οποία εκτελεί το query που της δίνεται. Επειδή πρέπει να εξασφαλιστεί ότι δεν θα τρέξει κάποια εντολή Insert ή Update στην βάση της Oracle, γι’ αυτό η μόνη εντολή που μπορεί να εκτελέσει η μέθοδος Execute είναι η εντολή Select. Γενικά από την Oracle αντλούμε μόνο δεδομένα και δεν κάνουμε καμία τροποποίηση ή προσθήκη. Τα αποτελέσματα της εκτέλεσης της εντολής αποθηκεύονται στις μεταβλητές μέλη της κλάσης.

6.6.5 OracleDatabaseQueries.java

Σε αυτή την κλάση βρίσκονται συγκεντρωμένες όλες οι μέθοδοι οι οποίες αντλούν δεδομένα από την βάση της Oracle.

Συγκεκριμένα:

- getNewCustomers(String webCustStr): Η μέθοδος αυτή επιστρέφει τον κωδικό της εταιρείας, την επωνυμία, το email του υπευθύνου, το όνομα, το επίθετο και την περιοχή της επιχείρησης των πελατών που έχουν τουλάχιστον ένα ενεργό agreement στην βάση της Oracle και ταυτόχρονα οι πελάτες αυτοί δεν υπάρχουν καταχωρημένοι στην βάση της MSSQL (ενεργοί ή μη ενεργοί). Επιστρέφει έναν Vector όπου στην κάθε του θέση υπάρχει ένα αντικείμενο τύπου Company.
- getCustomerKeywords(Company company): Επιστρέφει για συγκεκριμένο πελάτη, τις λέξεις κλειδιά που παρακολουθεί και τον κωδικό της κάθε λέξης κλειδί (για τις λέξεις κλειδιά που ανήκουν σε ενεργό agreement και μόνο).
- getActiveCustomersKeywords(String webCustStr): Επιστρέφει έναν Vector ο οποίος περιέχει αντικείμενα τύπου Company. Η κάθε εταιρεία που καταχωρείται στον Vector έχει τουλάχιστον ένα ενεργό agreement και ταυτόχρονα είναι καταχωρημένη στην βάση της MSSQL.
- getCustomersToActivate(String webCustrStr): Επιστρέφει έναν πίνακα τύπου String όπου αποθηκεύονται οι κωδικοί των πελατών που χρειάζεται να ενεργοποιηθούν. Ένας κωδικός χρειάζεται να ενεργοποιηθεί όταν έχει τουλάχιστον ένα agreement ενεργό στην βάση της Oracle αλλά στην βάση της MSSQL ο πελάτης είναι σταματημένος (δλδ υπάρχει στο πεδίο contract_expiration_date συμπληρωμένη η ημερομηνία στην οποία έχει σταματήσει ο πελάτης). Αυτή η μέθοδος επιστρέφει αποτελέσματα μόνο αν υπάρχουν πελάτες οι οποίοι σταμάτησαν και στην συνέχεια ξεκίνησαν ξανά την συνδρομή τους.
- getCustomersToDeactivate(String webCustStr): Επιστρέφει έναν πίνακα τύπου String όπου αποθηκεύονται οι κωδικοί των πελατών που χρειάζεται να απενεργοποιηθούν. Για να χρειάζεται ένας λογαριασμός απενεργοποίηση σημαίνει ότι ο πελάτης αυτός δεν έχει κανένα ενεργό

agreement στην Oracle ενώ στην βάση της MSSQL, το πεδίο contract_expiration_date είναι NULL (δλδ ο πελάτης είναι ενεργός).

6.7 Αρχιτεκτονική Producer – Consumer και μετατροπή αρχείου από PDF σε DjVu.

Η παραγωγή των αρχείων PDF και XML από το υποσύστημα παραγωγής δημιουργεί μία συνεχόμενη ροή αρχείων προς επεξεργασία για την εφαρμογή. Όπως έχουμε αναφέρει τα αρχεία αυτά αποθηκεύονται σε ένα φάκελο και περιμένουν η εφαρμογή να τα χειριστεί. Χρειάζεται ένα thread το οποίο θα ορίζει με ποια σειρά θα επεξεργάζονται τα αρχεία και πολλά threads (όσα ορίζει το Configuration.xml) τα οποία θα επεξεργάζονται τα αρχεία. Κάθε thread θα επεξεργάζεται και από μία δυάδα αρχείων (PDF και XML) τη φορά. Αυτή είναι η λεγόμενη αρχιτεκτονική «Producer – Consumer", δηλαδή «Παραγωγού – Καταναλωτή». Ο λόγος που επιλέχθηκε αυτή η αρχιτεκτονική είναι επειδή δεν μπορούν τα αρχεία προς επεξεργασία να «μοιραστούν» στα επιμέρους threads που θα τα επεξεργάζονται αφού η ολοκλήρωση της επεξεργασίας δεν έχει σταθερά ορισμένο χρόνο αλλά ποικίλει με βάση το μέγεθος του PDF αρχείου. Η διαδικασία η οποία δημιουργεί αυτή την καθυστέρηση είναι η μετατροπή του αρχείου PDF σε DJVU. Ένα PDF πέντε σελίδων θα πάρει πολύ περισσότερο χρόνο στην μετατροπή του από ένα αντίστοιχο μονοσέλιδο αρχείο. Κατά επέκταση δεν μπορεί να προβλεφθεί ο χρόνος που χρειάζεται για να ολοκληρωθεί η μετατροπή. Αν δεν υπήρχε αυτό το πρόβλημα θα μπορούσαμε να πούμε απλά ότι αν όλα τα αρχεία προς επεξεργασία την χρονική στιγμή t είναι 100 και έχουμε 5 threads, θα δίναμε παραμετρικά τα paths των 20 πρώτων στο πρώτο thread, τα 20 επόμενα στο δεύτερο κλπ. Στην περίπτωσή μας όμως, αν το τρίτο thread τελείωνε πρώτο, θα έπρεπε να περιμένει να τελειώσουν και όλα τα υπόλοιπα, γιατί τότε θα γίνει ο επανακαταμερισμός των αρχείων στα threads. Σκοπός της εφαρμογής είναι να μην χρονοτριβεί. Δηλαδή με το που τελειώνει ένα thread την επεξεργασία μιας δυάδας αρχείων πρέπει να προχωρήσει στην επεξεργασία της ακριβώς επόμενης.

Δουλειά του producer thread είναι να πάρει τα περιεχόμενα του φακέλου όπου βρίσκονται τα αρχεία προς επεξεργασία και να τα ταξινομήσει με βάση την ημερομηνία τροποποίησης του PDF αρχείου (προηγούνται τα παλαιότερα). Στην συνέχεια, παίρνει τα «σταθερά» επί «αριθμός threads» παλαιότερα αρχεία και αποθηκεύει τα ονόματά τους σε ένα java.util.concurrent.ArrayBlockingQueue.

Η ArrayBlockingQueue είναι μία δομή ουράς FIFO (First In First Out) σε μορφή πίνακα με σταθερό μέγεθος. Επίσης έχει την ιδιότητα να μπλοκάρει τα threads τα οποία θέλουν να πάρουν στοιχεία από αυτήν στην περίπτωση που είναι άδεια, όπως και να μπλοκάρει τα threads τα οποία θέλουν να εισάγουν κάτι σε αυτήν ενώ δεν υπάρχει διαθέσιμη θέση. Η ArrayBlockinQueue εξασφαλίζει ότι η χρήση της είναι ασφαλής όταν χρησιμοποιείται σε multithreaded εφαρμογές. Το μέγεθος αυτής της ουράς είναι «σταθερά» επί «αριθμός threads». Κάθε φορά που αδειάζει, το produce thread φροντίζει να εισάγει τα επόμενα ονόματα αρχείων προς επεξεργασία. Αν δεν υπάρχουν αρχεία προς επεξεργασία, θα πρέπει να ελέγχει ανά τακτά χρονικά διαστήματα (το πόσο συχνά ορίζεται στο Configuration.xml) έτσι ώστε όταν μπουν τα πρώτα αρχεία να αρχίσει άμεσα η επεξεργασία τους. Αν από την άλλη μεριά, τα αρχεία είναι υπερβολικά πολλά, με βάση τις ρυθμίσεις του Configuration.xml, έχει την δυνατότητα να μεταφέρει όλα τα αρχεία τα οποία δεν μπορεί να διαχειριστεί σε άλλη τοποθεσία έτσι ώστε να επεξεργαστούν από την αντίστοιχη εφαρμογή, η οποία αποτελεί «κλώνο» της ίδιας.

Η ουρά είναι ο συνδετικός κρίκος μεταξύ του producer thread και των consumer threads. Τα consumer threads από την άλλη, κατά την έναρξη της λειτουργίας τους ή μετά την ολοκλήρωση της προηγούμενης επεξεργασίας, παίρνουν από την ουρά το επόμενο αρχείο προς επεξεργασία. Έτσι, το κάθε αρχείο επεξεργάζεται από το πρώτο thread το οποίο θα είναι διαθέσιμο. Η επεξεργασία του αρχείου συμπεριλαμβάνει την μετατροπή του σε Djvu, την εισαγωγή των δεδομένων του XML αρχείου στην βάση και την τοποθέτηση των PDF, Djvu και XML αρχείων στην τελική τους θέση έτσι ώστε να είναι προσβάσιμα μέσω του ιστότοπου της εταιρείας από τους πελάτες.

Η μετατροπή του αρχείου γίνεται με χρήση command line εντολών τις οποίες καλεί η εφαρμογή. Οι εντολές αυτές με την σειρά τους μετατρέπουν το

αρχείο. Υπάρχουν δύο software τα οποία χρησιμεύουν για την μετατροπή. Στην περίπτωση που αποτύχει το πρώτο, εκτελείται το δεύτερο.

Επιπλέον, τα consumer threads πρέπει να αφήνουν log file σχετικά με την πορεία της επεξεργασίας και να στέλνουν email κατά περίπτωση.

Στην συνέχεια, ακολουθεί η ανάλυση των κλάσεων που συμβάλουν στην πιο πάνω λειτουργικότητα.

6.7.1 MoveInputFiles.java

Η κλάση αυτή αναλαμβάνει την μεταφορά των αρχείων PDF και XML στην περίπτωση που το πλήθος των διαθέσιμων αρχείων υπερβαίνει το όριο το οποίο έχει τεθεί στο ConfigurationFile.xml και είναι το attribute «ifFilesMoreThan». Το κέρδος από αυτή την λογική είναι η δυνατότητα του να τρέχουν πολλά αντίγραφα τις εφαρμογής ταυτόχρονα σε διαφορετικά PC/Servers μοιράζοντας τον φόρτο ανάλογα με τις δυνατότητες του κάθε μηχανήματος. Τα αρχεία θα μεταφέρονται σε μία άλλη διεύθυνση η οποία ορίζεται στο element «redirectionPath». Ένα αντίγραφο της εφαρμογής η οποία θα έχει πρόσβαση σε αυτή την νέα διεύθυνση θα επεξεργάζεται τα αρχεία που έχουν μεταφερθεί. Το αντίγραφο της εφαρμογής θα έχει στο Configuration.xml αρχείο στο element «general_input» την ίδια διεύθυνση που έχει η «πρωτεύουσα εφαρμογή» στο «redirectionPath». Συνοψίζοντας, η μία εφαρμογή, μεταφέρει τα επιπλέον αρχεία τα οποία αδυνατεί να χειριστεί σε μία άλλη διεύθυνση η οποία αποτελεί είσοδο για το «κλώνο» της εφαρμογής. Δεν είναι εφικτή η απενεργοποίηση αυτής της λειτουργίας από το αρχείο παραμετροποίησης, όμως αν ουσιαστικά οριστεί ο αριθμός του attribute “IfFilesMoreThan” υπερβολικά μεγάλος, θα έχει ως αποτέλεσμα να μην μεταφερθεί ποτέ κάτι.

6.7.2 ProducerThread.java

Η ProducerThread.java είναι ένα thread το οποίο κατευθύνει τα υπόλοιπα thread ως προς το ποια αρχεία θα επεξεργαστούν. Το μέλος inqueue είναι

τύπου ArrayBlockingQueue<Integer> και ως τέτοιο έχεις τις ακόλουθες ιδιότητες:

- Έχει συγκεκριμένο μέγεθος το οποίο ορίζεται κατά την δημιουργία του.
- Αν κάποιο thread προσπαθήσει να πάρει κάποιο στοιχείο από το inqueue ενώ είναι άδειο, θα γίνει block, δηλαδή θα ανασταλεί η λειτουργία του μέχρι να υπάρξει διαθέσιμο στοιχείο. Αν κάποιο thread θελήσει να εισάγει κάποιο στοιχείο στο inqueue ενώ είναι γεμάτο, θα ανασταλεί η λειτουργία του μέχρι να απελευθερωθεί μία κενή θέση.
- Οι εισαγωγές και εξαγωγές των στοιχείων του inqueue γίνονται με την λογική FIFO. Δηλαδή όταν ένα thread ζητήσει κάποιο δεδομένα, η ουρά θα επιστρέψει το παλαιότερο στοιχείο της το οποίο είναι η κεφαλή της ουράς. Αν κάποιο thread θελήσει να τοποθετήσει κάποιο στοιχείο, αυτό θα τοποθετηθεί στο τέλος της ουράς. Τα ενδιάμεσα στοιχεία δεν είναι προσβάσιμα.

Η inqueue υλοποιηεί την συλλογή (collection) Integer, και άρα τα περιεχόμενά της αναγκαστικά θα αποτελούνται από ακέραιους αριθμούς. Ο κάθε αριθμός αποτελείται από το όνομα του αρχείου PDF χωρίς την κατάληξη. Το ίδιο όνομα (αλλά με διαφωρετική κατάληξη) έχει και το XML αρχείο. Κατά την εκκίνησή του το thread ελέγχει αν η inqueue είναι άδεια. Αν όχι, περιμένει μέχρι να αδειάσει. Αν ναι, προχωράει στην εισαγωγή των ακεραίων στην inqueue. Ελέγχει αν υπάρχουν αρχεία προς επεξεργασία. Αν όχι, περιμένει για λίγο μέχρι να κάνει λίγο αργότερα τον επόμενο έλεγχο. Το διάστημα ανά το οποίο ελέγχει τον φάκελο για περιεχόμενα ορίζεται στο Configuration.xml στο attribute “interval”. Όταν βρεθούν αρχεία στην διεύθυνση καλείται η MoveInputFiles.java η οποία καθορίζει αν πρέπει να μεταφερθούν αρχεία και σε άλλο αντίγραφο της εφαρμογής ή όχι. Τέλος, εισάγει τα περιεχόμενα τα οποία βρήκε στο inqueue. Αξίζει να τονίσουμε πως η σειρά με την οποία μπαίνουν τα ονόματα των αρχείων στο inqueue ορίζεται από την κλάση CheckAndGetSortedList.java, και είναι η ημερομηνία τροποποίησης.

6.7.3 ProductionFiles.java

Η κλάση αυτή «χτίζει» για όλα τα αρχεία εισόδου όλες τις πιθανές διευθύνσεις των αρχείων (όπως και για το Djvu αρχείο) έτσι ώστε οι ενέργειες

επί των αρχείων (όπως οι μεταφορές) να είναι εύκολα διαχειρήσιμες. Οι διευθύνσεις αυτές ορίζονται από το αρχείο παραμετροποίησης και είναι η διεύθυνση εισόδου, η διεύθυνση εξόδου και η διεύθυνση εξόδου σε περίπτωση λάθους.

Για παράδειγμα, έστω τα αρχεία 123456.PDF, 123456.XML και έστω ότι η εφαρμογή διαβάζει τα αρχεία αυτά από την διεύθυνση «C:\INPUT». Έστω ότι η έξοδος (εκεί που καταλήγουν τελικά τα αρχεία) είναι η διεύθυνση «C:\wwwroot\inetpub\OUTPUT» και στην περίπτωση που κάτι δεν πάει κατά την επεξεργασία, έστω ότι καταλήγουν στη διεύθυνση «C:\onError».

Η productionFiles.java θα καταχωρήσει στον πίνακα generalInputFiles, τύπου String, τις ακόλουθες διευθύνσεις ως String: «C:\INPUT\123456.XML», «C:\INPUT\123456.PDF» και «C:\INPUT\123456.DJVU». Στον αντίστοιχο πίνακα onErrorFiles θα αποθηκεύσει τα String «C:\onError\123456.PDF», «C:\onError\123456.XML» και «C:\onError\123456.DJVU».

Στον πίνακα destinationFiles αποθηκεύονται όμοια οι αντίστοιχες διευθύνσεις αλλά με διαφορετική λογική. Καταρχάς τα αρχεία αποθηκεύονται με βάση την κατάληξή τους σε διαφορετικούς υποφακέλους. Τα PDF αρχεία θα αποθηκευθούν στο «C:\wwwroot\inetpub\OUTPUT\PDF», τα DjVu στο «C:\wwwroot\inetpub\OUTPUT\DJVU» και τα XML «C:\wwwroot\inetpub\OUTPUT\XML».

Στην συνέχεια, τα PDF και τα Djvu ακολουθούν την ακόλουθη σύμβαση. Τα αρχεία μοιράζονται σε φακέλους με βάση το όνομα του αρχείου, και συγκεκριμένα στο φάκελο που έχει όνομα ίσο με το πηλίκο της διαίρεσης του ονόματος του αρχείου με το 200. Στο παράδειγμά μας, το αρχείο 123456.PDF θα αποθηκευθεί στο φάκελο 617 (123456 mod 200 = 617). Αυτό έχει ως αποτέλεσμα ο κάθε φάκελος να έχει το πολύ 200 αρχεία. Έτσι, το PDF θα πρέπει να καταχωρηθεί στην διεύθυνση «C:\wwwroot\inetpub\OUTPUT\PDF\617\123456.PDF» και το αντίστοιχο Djvu, το οποίο θα έχει παραχθεί, στην διεύθυνση «C:\wwwroot\inetpub\OUTPUT\DJVU\617\123456.DJVU».

Για τα XML ακολουθείτε άλλη σύμβαση, και συγκεκριμένα καταχωρούνται ημερολογιακά. Αν δηλαδή το αρχείο επεξεργάστηκε την 1/2/2011, αυτό θα πρέπει να καταχωρηθεί στην διεύθυνση «C:\wwwroot\inetpub\OUTPUT\XML\1_2_2011\123456.XML». Όλες αυτές οι

διευθύνσεις «χτίζονται» κατά την δημιουργία αντικειμένου τύπου ProductionFiles μέσο των μεθόδων setGeneralInputFiles(), setOnErrorFiles() και setDestFiles().

Ερχόμενοι στις λειτουργίες της κλάσης, ακολουθεί η λειτουργία του ελέγχου των αρχείων. Ο έλεγχος αυτός περιλαμβάνει τον έλεγχο ύπαρξης των αρχείων. Όταν η εφαρμογή παίρνει τα περιεχόμενα του φακέλου εισόδου, το κάνει μόνο για τα αρχεία τύπου PDF. Έτσι, η εφαρμογή πρέπει να ελέγξει αν υπάρχει και το αντίστοιχο XML στον φάκελο πριν προχωρήσει σε οποιαδήποτε επεξεργασία. Αν υπάρχουν αυτά τα δύο αρχεία, τότε η μέθοδος getActionOption() θα επιστρέψει τον αριθμό 2, που σημαίνει πως υπάρχουν δύο αρχεία εισόδου και είναι το PDF μαζί με το αντίστοιχο XML. Η εφαρμογή αναπτύχθηκε έτσι ώστε να υποστηρίζει ακόμα μία περίπτωση, την περίπτωση όπου στον φάκελο εισόδου υπάρχει το PDF, XML και το Djvu. Σε αυτή την περίπτωση η μέθοδος getAcrionOption() θα επιστρέψει τον αριθμό 3. Αυτή η περίπτωση προκύπτει μόνο υπό το ακόλουθο σενάριο. Έστω ότι έχουμε ένα αρχείο PDF μαζί με το XML του, αλλά το XML περιέχει κάποια λάθη τα οποία δεν επιτρέπουν να ολοκληρωθεί η επεξεργασία του. Σε αυτή την περίπτωση αφού η εφαρμογή δεν θα μπορέσει να διεκπεραιώσει την επεξεργασία της εισαγωγής των πληροφοριών του XML, αλλά θα έχει μετατρέψει επιτυχώς το PDF σε Djvu, θα πρέπει να μεταφέρει τα τρία αυτά αρχεία στην διεύθυνση «C:\onError» και θα αποστείλει ένα email το οποίο θα ενημερώσει το τμήμα μηχανογράφησης για το πρόβλημα του αρχείου. Αφού το XML αρχείο διορθωθεί από το τμήμα, θα πρέπει να το τοποθετήσουν εκ νέου μαζί με το PDF και το ήδη δημιουργημένο DjVu στον φάκελο εισόδου έτσι ώστε να εισαχθεί τελικά από την εφαρμογή. Ουσιαστικά εισάγοντας και το Djvu η εφαρμογή γλυτώνει χρόνο επειδή δεν μπαίνει εκ νέου στην διαδικασία να μετατρέψει το PDF σε Djvu.

Η τελευταία βασική λειτουργία της κλάσης είναι η μεταφορά των αρχείων μέσω της μεθόδου moveFiles(int option, int ConvertResult, bookean successfulUpdate). Συγκεκριμένα, αν η μετατροπή του PDF αρχείου είναι επιτυχής και η ενημέρωση της βάσης είναι επιτυχής, τα αρχεία πρέπει να μεταφερθούν στις τελικές τους θέσεις. Η δημιουργία του Djvu αρχείου γίνεται απευθείας στην τελική του θέση. Αν κάτι από τα δύο δεν ολοκληρωθεί επιτυχώς, όλα τα αρχεία πρέπει να μεταφερθούν στο OnError. Αντίστοιχα, αν

στην είσοδο έχουμε τρία αρχεία και η εισαγωγή του XML ολοκληρωθεί επιτυχώς θα πρέπει να μεταφερθούν στην τελική τους διεύθυνση, ενώ διαφορετικά στον OnError.

6.7.4 Convert.java

Η Convert.java υλοποιεί την μετατροπή των αρχείων PDF σε Djvu. Για την επίτευξη αυτής της διαδικασίας πρακτικά χρησιμοποιείται εξωτερικό software το οποίο κάνει την μετατροπή. Η εφαρμογή χρησιμοποιεί αυτό το software μέσω command line εντολών έτσι ώστε να αυτοματοποιηθεί αυτή η διαδικασία και να μπορεί να είναι συγχρονισμένη με τις υπόλοιπες λειτουργίες της εφαρμογής. Δεν πρέπει δηλαδή να επιτρέπεται να εισαχθεί κάποιο απόκομμα στην βάση αλλά το Djvu αρχείο να μην έχει ακόμα δημιουργηθεί. Όταν η βάση ενημερώνεται πρέπει όλα τα αρχεία να είναι στη θέση τους για να είναι διαθέσιμα στον χρήστη μέσο του ιστότοπου της εταιρείας.

Για την μετατροπή χρησιμοποιούνται δύο software. Υπάρχουν αρχεία PDF τα οποία δεν μπορούν να μετατραπούν από το πρώτο αλλά μετατρέπονται με επιτυχία από το δεύτερο. Έτσι, μόνο στην περίπτωση που το πρώτο software αποτύχει, η εφαρμογή χρησιμοποιεί το δεύτερο. Ο λόγος που η εφαρμογή δεν χρησιμοποιεί αποκλειστικά το δεύτερο είναι επειδή δεν διαθέτει τις ίδιες δυνατότητες στην παραμετροποίηση όσον αφορά τα τεχνικά χαρακτηριστικά του εξερχόμενου Djvu, με αποτέλεσμα τα παραγόμενα αρχεία να έχουν μεγαλύτερο μέγεθος.

Για να μπορέσουμε να καλέσουμε τις εφαρμογές αυτές μέσα από τον κώδικά μας χρησιμοποιούμε την κλάση java.lang.ProcessBuilder.java η οποία χρησιμοποιείται για την δημιουργία διεργασιών στο λειτουργικό σύστημα. Η εφαρμογή, δημιουργεί ένα ProcessBuilder αντικείμενο δίνοντας στο δομητή ένα πίνακα από String ο οποίος περιέχει όλα τα ορίσματα για την εκτέλεση της εντολής. Τα ορίσματα αυτά ορίζονται στο αρχείο παραμετροποίησης. Έτσι, ο πίνακας των ορισμάτων θα μπορούσε για παράδειγμα να είναι ο String[] commandArray = {“documenttodjvu.exe”,”—profile=convert”,”C:/INPUT/123456.PDF”,”«C:\wwwroot\inetpub\OUTPUT\DjVu\123456.DJVU”}. Δηλαδή δίνεται το εκτελέσιμο πρόγραμμα, το προφίλ «convert» το οποίο είναι ρυθμισμένο/αποθηκευμένο στο πρόγραμμα

μετατροπής, η διεύθυνση του αρχείου που θέλουμε να γίνει η μετατροπή όπως και η διεύθυνση όπου θέλουμε να δημιουργηθεί το djvu αρχείο. Καλώντας την μέθοδο start() ξεκινάει η εκτέλεση της εντολής. Η εφαρμογή δεν ξέρει πότε θα τελειώσει η εκτέλεση της εντολής και δεν πρέπει να συνεχίσει την εκτέλεση του κώδικα που ακολουθεί πριν να είναι διαθέσιμο το αποτέλεσμα της μετατροπής. Έτσι καλώντας την waitFor() το thread αδρανοποιείται μέχρι να ολοκληρωθεί η εκτέλεση της εντολής. Μετά την ολοκλήρωσή της, μέσω της exitValue() παίρνουμε την τιμή με την οποία τερμάτισε η εντολή.

Αν η μετατροπή δεν πραγματοποιηθεί με επιτυχία (δηλαδή η τιμή που επιστρέφεται είναι διάφορη του μηδενός) πολλές φορές μένει ένα αρχείο djvu, μηδενικού μεγέθους, στην διεύθυνση που έχουμε ορίσει και το οποίο πρέπει να διαγραφεί. Στην συνέχεια η εφαρμογή θα προσπαθήσει να κάνει την μετατροπή μέσο του δεύτερου προγράμματος εκτελώντας με όμοιο τρόπο την αντίστοιχη εντολή. Αν κανένα πρόγραμμα δεν καταφέρει να κάνει επιτυχώς την μετατροπή στέλνεται ένα ενημερωτικό email προς το τμήμα μηχανογράφισης και το μέλος exitValue έχει την τιμή -1.

Αν τρέξουμε το κάθε ένα πρόγραμμα μέσο γραμμής εντολών, εμφανίζεται στο DOS παράθυρο ένα κείμενο το οποίο δίνει πληροφορίες για την διαδικασία του convert καθώς αυτή πραγματοποιείται όπως και τα μηνύματα λάθους που μπορεί να προκύψουν. Προκειμένου να είναι αυτές οι πληροφορίες διαθέσιμες στην εφαρμογή, ανακατευθύνουμε την έξοδο των προγραμμάτων σε log αρχείο, και συγκεκριμένα στο ConvertOutputLog.txt. Το log αυτό αρχείο γίνεται overwrite κάθε φορά που γίνεται μία μετατροπή. Μόνο στην περίπτωση που μία μετατροπή αποτύχει, το περιεχόμενο του ConvertOUtputLog.txt που έχει δημιουργηθεί αντιγράφεται στο log αρχείο log.txt έτσι ώστε να είναι διαθέσιμο αν χρειαστεί.

Τέλος, να σημειώσουμε ότι η java.lang.ProcessBuilder.Redirect η οποία χρησιμοποιείται για την ανακατεύθυνση της εξόδου του DOS παράθυρου στο αρχείο ConvertOutputLog.txt είναι διαθέσιμη μόνο στο JDK7.

6.7.5 UpdateDatabase.java

Η κλάση αυτή χρησιμοποιεί τις κλάσεις που έχουμε ήδη αναφέρει έτσι ώστε να διαβάσει το XML αρχείο που έχει παραχθεί από το υποσύστημα παραγωγής και στη συνέχει να εισάγει τα δεδομένα του στην βάση. Διαβάζει το XML μέσω της ReadDistributionXML.java και εισάγει τα δεδομένα του μέσω της SQLDatabaseQueries.java. Αν οι διαδικασίες ολοκληρωθούν φυσιολογικά, το μέλος updateResult θα έχει την τιμή true, διαφορετικά θα έχει την τιμή false.

6.7.6 Consumer Thread.java

Η ConsumerThread.java είναι ένα thread το οποίο διαχειρίζεται την επεξεργασία των αποκομμάτων. Κάθε thread επεξεργάζεται μία δυάδα αρχείων PDF, XML τη φορά. Παίρνει από την inqueue (τύπου ArrayBlockingQueue) το όνομα του αρχείου που έχει σειρά να επεξεργαστεί. Στην συνέχεια ελέγχει αν υπάρχουν και τα αντίστοιχα XML και DJVU αρχεία μαζί με το PDF (κλάση ProductionFiles). Αν είναι μόνο δύο τα αρχεία στον φάκελο εισόδου προχωράει στην μετατροπή του PDF σε DJVU (Convert.java) και εφ' όσον ολοκληρωθεί με επιτυχία (με τη βοήθεια οποιοδήποτε από τα δύο εξωτερικά software), προχωράει στην ενημέρωση της βάσης. Αν από την άλλη υπάρχουν και τα τρία αρχεία στην διεύθυνση εισόδου, παραλείπει το βήμα της μετατροπής.

Τέλος, στην περίπτωση που η μετατροπή του αρχείου δεν ολοκληρώθηκε επιτυχώς, η έξοδος του προγράμματος αντιγράφεται από το ConvertOutputLog.txt στο Log.txt.

6.7.7 AccountsSynch.java

Η AccountsSynch.java υλοποιεί την κλάση java.util.TimerTask η οποία αποτελεί μία διεργασία με δυνατότητα προγραμματισμού της εκτέλεσής της είτε μόνο μία φορά είτε περιοδικά. Η διεργασία που θέλουμε να υλοποιηθεί

στην περίπτωσή μας είναι η ενημέρωση της βάσης του υποσυστήματος δημοσίευσης σε σχέση με την βάση του υποσυστήματος παραγωγής. Το πόσο συχνά θα πραγματοποιείται αυτή η διεργασία δίνεται στο αρχείο παραμετροποίησης μέσο του element <interval>, που είναι η περίοδος εκφρασμένη σε ώρες. Αν δηλαδή υπάρχει ο αριθμός 24, αυτό συμβαίνει ότι θα τρέχει τον συγχρονισμό μία φορά την ημέρα. Η ώρα που θα πραγματοποιείται ο συγχρονισμός είναι επίσης μια σημαντική παράμετρος. Αυτή δίνεται μέσα από το element <start_time> στο αρχείο παραμετροποίησης εκφρασμένο σε ώρες σε 24ωρη μορφή. Τέλος, υπάρχει η δυνατότητα ενεργοποίησης ή απενεργοποίησης της λειτουργίας του συγχρονισμού μέσα από το attribute <enabled>. Αυτό είναι χρήσιμο σε περιπτώσεις όπου τρέχουν πολλοί κλώνοι της εφαρμογές σε διαφορετικά μηχανήματα και χρειάζεται μόνο ένα να τρέχει τον συγχρονισμό.

Η AccountsSynchr.java κατά την εκτέλεσή της αρχικά ελέγχει για νέους πελάτες οι οποίοι πρέπει να καταχωρηθούν. Για να βρεθούν οι νέοι πελάτες απαιτείται η εκτέλεση δύο ερωτημάτων. Ένα προς τη βάση δημοσίευσης για να βρεθούν όλοι οι ενεργοί πελάτες και ένα δεύτερο ερώτημα προς την παραγωγική βάση για να φέρουμε όλους τους ενεργούς πελάτες οι οποίοι δεν υπάρχουν στην βάση δημοσίευσης. Με την εκτέλεση του δεύτερου ερωτήματος πρακτικά έχουμε όλους τους νέους πελάτες. Στην συνέχεια δημιουργούνται αυτοί οι πελάτες εισάγοντας τα στοιχεία της εταιρείας, έναν αρχικό λογαριασμό για το site και τις λέξεις κλειδιά που ενδιαφέρουν τον πελάτη. Μετά την δημιουργία του πελάτη, η εφαρμογή αποστέλλει το ενημερωτικό email στον πελάτη για την έναρξη του λογαριασμού του. Τέλος, καταγράφει ποιους πελάτες ενεργοποίησε ή δεν κατάφερε να ενεργοποιήσει στο Synchronization_log.txt.

Το δεύτερο κατά σειρά είναι η ενεργοποίηση των πελατών που απενεργοποιήθηκαν κάποια στιγμή στο παρελθόν και πλέον είναι ενεργοί. Όμοια, για να βρεθούν αυτοί οι πελάτες χρειάζεται να εκτελεστούν δύο ερωτήματα. Το πρώτο ερώτημα απευθύνεται στην βάση του υποσυστήματος δημοσίευσης και φέρνει όλους τους πελάτες που δεν είναι πλέον ενεργοί. Το δεύτερο ερώτημα εκτελείται στην βάση του παραγωγικού υποσυστήματος και επιστρέφει όλους τους πελάτες που είναι ενεργοί και ταυτόχρονα είναι απενεργοποιημένοι στην βάση του υποσυστήματος δημοσίευσης. Στην

συνέχεια ενεργοποιεί τους πελάτες αυτούς ορίζοντας για τους πελάτες αυτούς NULL στο πεδίο contract_expiration_date. Τέλος καταγράφει την επιτυχημένη ή όχι ενεργοποίηση του εκάστοτε πελάτη.

Στη συνέχεια του συγχρονισμού πραγματοποιείται η απενεργοποίηση των πελατών που σταματούν την συνδρομή τους. Και πάλι, το πρώτο ερώτημα παίρνει όλους τους ενεργούς πελάτες από την βάση του υποσυστήματος δημοσίευσης ενώ το δεύτερο ερώτημα όλους τους ανενεργούς πελάτες από την βάση του υποσυστήματος παραγωγής και οι οποίοι πελάτες είναι ενεργοί στην βάση του υποσυστήματος δημοσίευσης, και άρα πρέπει να απενεργοποιηθούν. Στην συνέχεια γίνεται η απενεργοποίηση για τους πελάτες αυτούς, εισάγοντας στο πεδίο contract_expiration_date την τρέχουσα ημερομηνία (timestamp). Τέλος καταγράφει την επιτυχημένη ή όχι απενεργοποίηση του εκάστοτε πελάτη.

Στην κλάση AccountsSynchr.java ακολουθεί ο συγχρονισμός των λέξεων κλειδιών όλων των ενεργών πελατών. Εδώ πρέπει να διευκρινίσουμε πως δεν θέλουμε να διαγράφονται οι λέξεις κλειδιά οι οποίες ο πελάτης αποφασίζει πως δεν τον ενδιαφέρουν πλέον και δεν θα λαμβάνει αποδελτίωση για αυτές, και αυτό γιατί θέλουμε να συνεχίσει να μπορεί να βλέπει το αρχείο των αποκομμάτων που διαθέτει ήδη.

Οι ενέργειες που μας ενδιαφέρουν είναι μόνο η δημιουργία των νέων λέξεων κλειδιών και ο συγχρονισμός των υφιστάμενων ως προς τυχόν αλλαγές που μπορεί να έχει υποστεί η ονομασία τους. Και οι δύο αυτές λειτουργίες υλοποιούνται κατά την χρήση του stored procedure sp_insertKeyword. Επιπλέον, το sp_insertKeyword δημιουργεί τους νέους πελάτες στην βάση στην περίπτωση που δεν υπάρχουν, και στην συνέχεια εισάγει τις λέξεις κλειδιά για τις οποίες ενδιαφέρεται. Γι' αυτό και το sp_insertKeyword χρησιμοποιείται από την AccountsSynchr.java κατά την δημιουργία νέων πελατών.

Η μόνη διαφορά που έχει η δημιουργία νέου πελάτη σε σχέση με τον συγχρονισμό των λέξεων κλειδιών είναι ότι στην πρώτη περίπτωση ο πελάτης θέλει να λαμβάνει ενημερωτικό email. Χρησιμοποιούμε δηλαδή την sp_insertKeyword και όταν θέλουμε να εισάγουμε νέο πελάτη και στην περίπτωση που θέλουμε να προσθέσουμε επιπλέον λέξεις κλειδιά σε κάποιον πελάτη.

Το sp_insertKeyword πριν εισάγει κάποιο keyword ελέγχει αν υπάρχει ο κωδικός του καταχωρημένος. Αν ναι, ελέγχει ότι το όνομα της λέξεις κλειδί είναι ίδιο με αυτό που έχει σκοπό να καταχωρήσει. Αν είναι ίδιο, δεν προβαίνει σε καμία ενέργεια. Διαφορετικά, τροποποιεί το όνομα της λέξεις κλειδί. Αν ο κωδικός της λέξης κλειδί δεν υπάρχει, τον εισάγει.

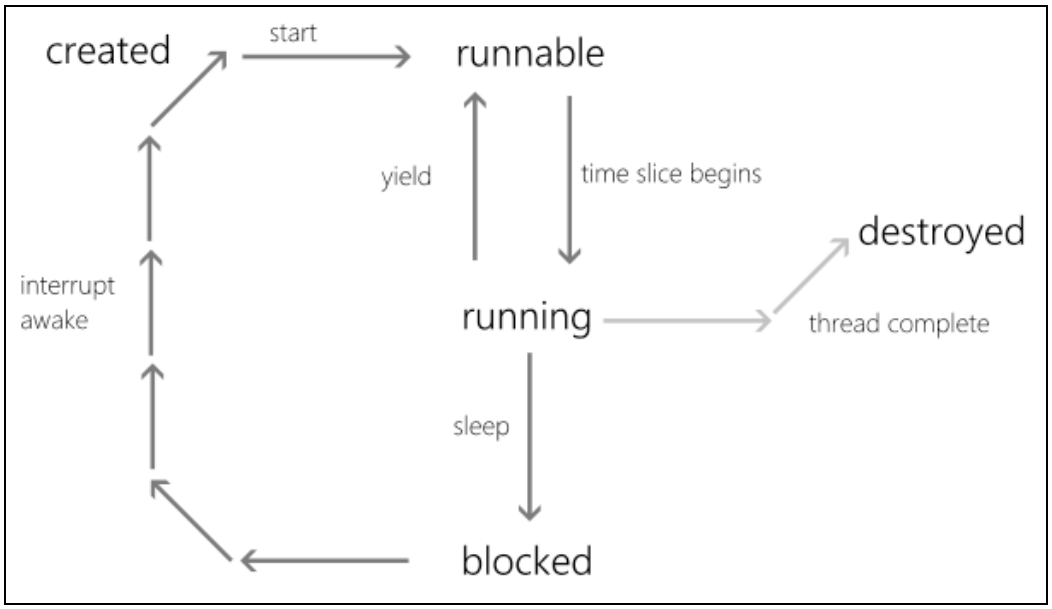
Πρακτικά η AccountsSynchr.java δίνει εντολή στην βάση να εισάγει ξανά όλους τους ενεργούς πελάτες (και κατά επέκταση όλες τις λέξεις κλειδιά), αλλά ο κώδικας του stored procedure προσθέτει μόνο όσα δεν υπάρχουν, δηλαδή τα νέα μόνο στοιχεία και τροποποιεί μόνο τα ονόματα των λέξεων κλειδιών που έχουν τροποποιηθεί.

Η μέθοδος createAccount της κλάσης SQLDatabaseQueries.java είναι αυτή η οποία εκτελεί το stored procedure στην βάση. Προκειμένου να διαχωριστεί το πότε η κλήση της μεθόδου αυτή έχει σκοπό των συγχρονισμό των λέξεων κλειδιών και πότε την δημιουργία νέων πελατών δίνουμε κατά την κλήση της createAccount την παράμετρο false ή true αντίστοιχα. Έτσι, μόνο στην δεύτερη περίπτωση θα αποσταλεί ενημερωτικό email έναρξης για τους πελάτες.

6.7.8 Start.java

Μέσω της Start κλάσης όλη η εφαρμογή τίθεται σε λειτουργία. Συγκεκριμένα, διαβάζει το αρχείο παραμετροποίησης, ρυθμίζει το Logging, δημιουργεί Producer thread, Consumer threads και τέλος δημιουργεί και ένα synchronization TimerTask. Ο αριθμός των Producer threads αντικειμένων ορίζεται στο αρχείο παραμετροποίησης.

Κάθε Thread μπορεί να περάσει από πολλές καταστάσεις όπως δείχνει η Εικόνα 22: Κύκλος ζωής των Threads. Τα βελάκια περιγράφουν τις μεθόδους οι οποίες αλλάζουν την κατάσταση του Thread.



Εικόνα 22: Κύκλος ζωής των Threads

Με την δημιουργία ενός αντικειμένου Thread το αντικείμενο αυτό είναι στην κατάσταση created.

```
ProducerThread prodThread = new ProducerThread(contHand, inqueue);
```

Κώδικας 19: *Δημιουργία ProducerThread*

Καλώντας την μέθοδο start() το thread γίνεται runnable, δηλαδή δίνουμε εντολή για την έναρξη της εκτέλεσής του από τον υπολογιστή. Από εκεί και πέρα εξαρτάται από τον υπολογιστή το πότε θα εκτελέσει το κώδικα του thread ανάλογα με την πολιτική που ακολουθεί για την διαχείριση όλων των εργασιών που πρέπει να εκτελέσει «ταυτόχρονα». Αν για παράδειγμα έχουμε έναν επεξεργαστή και ακολουθεί την πολιτική του time slice, αυτό θα σημαίνει ότι το thread θα εκτελείται για ένα συγκεκριμένο χρονικό διάστημα. Μετά από αυτό το διάστημα με βάση τις προτεραιότητες των υπόλοιπων διεργασιών σε αναμονή θα εκτελεστεί αυτή που έχει την μεγαλύτερη.

Την χρονική στιγμή που το thread εκτελείται από τον επεξεργαστή μεταβαίνει αυτόματα σε κατάσταση running. Όσο ο υπολογιστής εκτελεί άλλες εργασίες το thread ξαναγυρνάει αυτόματα στην κατάσταση runnable μέχρι να έρθει ξανά η σειρά του.

```
prodThread.start();
```

Κώδικας 20: *Μεταφορά του prodThread σε κατάσταση Runnable*

Όταν καλούμε την μέθοδο start() αυτόματα γίνεται κλήση της μεθόδου run(), την οποία κάνουμε implement, και μέσα σε αυτήν γράφουμε τον κώδικα τον οποίο θέλουμε να εκτελέσουμε. Άπαξ και τελειώσει η εκτέλεση του κώδικα στην μέθοδο run, το thread μεταβαίνει στην κατάσταση destroyed και μπορεί να συλλεχθεί ανά πάσα στιγμή από τον GC.

Όσο το thread δεν εκτελείται λόγω της μεθόδου sleep(long milis), βρίσκεται σε κατάσταση blocked. Μετά την πάροδο ορισμένων millisecond, το thread επιστρέφει και πάλι στην κατάσταση runnable.

Για να ενεργοποιηθεί η διαδικασία του synchronization χρειάζεται η δημιουργία ενός java.util.Timer και στην συνέχει να γίνει κλήση της μεθόδου scheduleAtFixedRate. Στην μέθοδο αυτή δίνονται παραμετρικά το TimerTask αντικείμενο, η ημερομηνία που πρέπει να ξεκινήσει το TimerTask και το διάστημα ανά το οποίο θα εκτελείται.

Η μέθοδος stopExecution() σταματάει την εκτέλεση όλων των threads καλώντας την interrupt() μέθοδο για το κάθε αντικείμενο. Η μέθοδος interrupt() δίνει εντολή να σταματήσει η εκτέλεση του thread, κάτι το οποίο δεν είναι απαραίτητο ότι θα γίνει άμεσα.

```
prodThread.interrupt();
```

Κώδικας 21: *Μεταφορά του prodThread σε κατάσταση Created*

Το κάθε thread έχει μια Boolean μεταβλητή η οποία ορίζει αν το thread είναι interrupted ή όχι. Όταν καλείται η μέθοδος interrupt() η μεταβλητή αυτή γίνεται true. Στις περιπτώσεις όπου το thread δεν τερματίζει ποτέ (όπως σε περιπτώσεις while(true){…}) και δεν υπάρχει η μέθοδος sleep θα πρέπει να ελέγχεται η μεταβλητή αυτή έτσι ώστε να σταματήσει η εκτέλεση του thread.

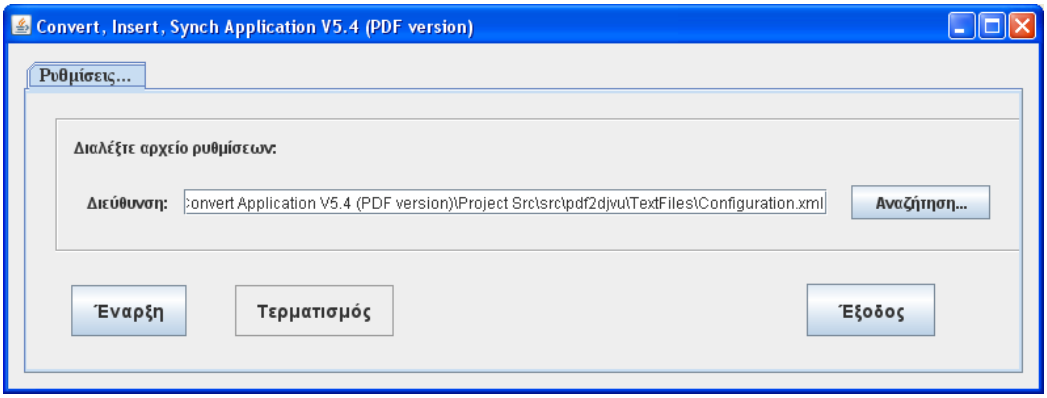
```
While (IThread.currentThread().isInterrupted() && more work to do)
{
   do more work
}
```

Κώδικας 22: (Παράδειγμα κώδικα) Περιοδικός έλεγχος για το αν το Thread έχει γίνει interrupt.

Στην εφαρμογή της παρούσας πτυχιακής δεν ελέγχεται ποτέ αυτή η μεταβλητή επειδή υπάρχει τουλάχιστον μία από τις μεθόδους sleep, take ή put σε αντίστοιχους βρόγχους επανάληψης. Αν το thread έχει γίνει interrupt, με το που θα κληθεί οποιαδήποτε από αυτές τις μεθόδους αυτόματα θα προκληθεί InterruptedException μετά το χειρισμό της οποίας σταματάει η εκτέλεση του thread.

6.7.9 ConvertApplicUI.java

Αυτή είναι η τελευταία κλάση της εφαρμογής και περιλαμβάνει το απαραίτητο γραφικό κομμάτι (βλ. *Εικόνα 23: Διεπιφάνεια εφαρμογής*) και λίγες γραμμές κώδικα για την έναρξη/διακοπή της λειτουργίας της επεξεργασίας των αποκομμάτων.



Εικόνα 23: *Διεπιφάνεια εφαρμογής*

Ξεκινώντας την περιγραφή, η μέθοδος setConfigFilePath() είναι αυτή η οποία διαβάζει το αρχείο ConfigurationLocation.txt, στο οποίο υπάρχει καταχωρημένη η διεύθυνση του Configuration.xml αρχείου. Η λογική πίσω από αυτό είναι ότι το ConfigurationLocation.txt αρχείο είναι πάντα

αποθηκευμένο σε συγκεκριμένη διεύθυνση και συγκεκριμένα στην ίδια διεύθυνση με αυτήν του εκτελέσιμου jar.

Η κλήση της μεθόδου setConfigFilePath() έχει εισαχθεί στο «Post-Creation Code» της καρτέλας Code των Properties του PathTxtField (χρησιμοποιώντας το GUI του NetBeans). Το PathTxtField είναι το TextField στο οποίο συμπληρώνεται αυτόματα (μέσω της setConfigFilePath()) το path στο οποίο βρίσκεται το αρχείο παραμετροποίησης.

Στη συνέχεια στην μέθοδο StartButCtionPerformed(java.awt.event.ActionEvent evt) δίνονται οι εντολές για την έναρξη της επεξεργασίας των αποκομμάτων. Συγκεκριμένα, παίρνει την διεύθυνση όπου είναι αποθηκευμένο το αρχείο παραμετροποίησης και το δίνει ως παράμετρο κατά την δημιουργία αντικειμένου της κλάσης Start.

Στη συνέχεια, στην μέθοδο StopButActionPerformed(java.awt.event.ActionEvent evt) δίνονται οι εντολές για τον τερματισμό της επεξεργασίας των αποκομμάτων. Καλείται η μέθοδος stopExecution() της κλάσης Start που έχει ως αποτέλεσμα όλα τα thread και ο TimerTask να σταματούν την εκτέλεσή τους.

6.8 Επίλογος

Σε αυτό το κεφάλαιο είδαμε την λογική του κώδικα, περιγράψαμε όλες τις κλάσεις και δώσαμε τα βασικά σημεία τους. Ταυτόχρονα, αναλύσαμε κάποιες κλάσεις της java οι οποίες ήταν απαραίτητες για την εφαρμογή, όπως η Thread, XMLReader, Logging, mail, ArrayBlockingQueue, ProcessBuilder κα. Ολόκληρος ο κώδικας της εφαρμογής δίνεται στο Παράρτημα Ε, σελ. 169.

Σε αυτό το σημείο ολοκληρώνεται το βασικό κομμάτι της πτυχιακής εργασίας. Στα παραρτήματα μπορείτε να βρείτε πλούσιο υλικό το οποίο συμπληρώνει και εμπλουτίζει την πτυχιακή.

Επίλογος

Επίτευξη στόχων

Συνοψίζοντας, μέσω της εφαρμογής που αναπτύχθηκε στην παρούσα πτυχιακή εργασία καλύφθηκαν όλες οι ανάγκες που απαιτούνταν για τον συγχρονισμό και ενημέρωση της βάσης του υποσυστήματος δημοσίευσης αντλώντας δεδομένα από την βάση του παραγωγικού υποσυστήματος. Ταυτόχρονα, δημιουργούνται και τα αρχεία DJVU από τα PDF αρχεία του παραγωγικού υποσυστήματος.

Η εφαρμογή, μέσω της παραμετροποίησης, δίνει ευελιξία ως προς το πόσο θα φορτώνεται ο Server στον οποίο είναι εγκατεστημένη και δίνει και την δυνατότητα να τρέχουν πολλαπλοί «κλώνοι» της εφαρμογής σε πολλούς Servers.

Σημαντικό κομμάτι αποτέλεσε και η αυτοματοποίηση, δηλαδή το να μην χρειάζεται κάποιος να επιβλέπει την διαδικασία, αλλά να ενημερώνεται και να λαμβάνει μέρος μόνο όταν κάτι δεν πήγε καλά στην ολοκλήρωση της διαδικασίας.

Η ασφάλεια είναι ένα σημείο το οποίο καλύφθηκε έτσι ώστε να μην είναι ευάλωτη η βάση από επιθέσεις τύπου SQL injections.

Η βάση σχεδιάστηκε με τέτοιο τρόπο έτσι ώστε να μπορεί ο οποιοσδήποτε (προγραμματιστής - developer) να την χρησιμοποιήσει εύκολα, χωρίς να εμπλακεί με τις λεπτομέρειες της λογικής της βάσης και ταυτόχρονα τα δεδομένα της βάσης δεν κινδυνεύουν από λάθος χειρισμούς.

Η καταγραφή των βημάτων επεξεργασίας από την εφαρμογή σε log files είναι ιδιαίτερα χρήσιμη για αναφορά και για την διαχείριση προβλημάτων.

Τέλος, πολύ σημαντικό είναι η εφαρμογή να μην αφήνει ανολοκλήρωτη οποιαδήποτε διαδικασία επεξεργασίας, από την στιγμή που αυτή έχει ξεκινήσει.

Προσθήκες

Όπως σε κάθε εφαρμογή, έτσι και για την εφαρμογή που διαπραγματεύεται η παρούσα πτυχιακή, υπάρχουν προσθήκες οι οποίες θα μπορούσαν να βελτιώσουν την εφαρμογή. Συγκεκριμένα, όσον αφορά την ασφάλεια, θα μπορούσε το πεδίο του password της MSSQL βάσης (δηλαδή της βάσης του υποσυστήματος δημοσίευσης) να υλοποιηθεί ως encrypted. Επιπρόσθετα, θα μπορούσαν να υλοποιηθούν πολλές λειτουργίες για την διαχείριση των αρχείων που δεν μπόρεσε να επεξεργαστεί η εφαρμογή. Τέτοιες λειτουργίες θα μπορούσε να ήταν:

- Σε δεύτερη καρτέλα στο GUI να υπάρχει ένα κουμπί στο οποίο όταν γίνεται click θα εμφανίζεται ένας πίνακας ο οποίος θα δίνει πληροφορίες για όλα τα αρχεία τα οποία δεν μπόρεσαν να επεξεργαστούν. Οι πληροφορίες θα περιλαμβάνουν το αν το απόκομμα είναι άρθρο ή δημοπρασία (και στην περίπτωση που είναι δημοπρασία θα εμφανίζει την ημερομηνία λήξης της δημοπρασίας και το ποσό – όπου συνήθως γίνονται τα λάθη), το αν η βάση έχει ενημερωθεί, το αν τα αρχεία (ένα ή περισσότερα) έχουν μεταφερθεί στις τελικές τους θέσεις, το αν η μετατροπή του PDF σε DJVU έγινε με επιτυχία κα.
- Ανάλογα με το πρόβλημα που αντιμετωπίζει το αρχείο το οποίο δεν μπόρεσε να επεξεργαστεί, η δεύτερη αυτή καρτέλα, να δίνει την δυνατότητα επιλογής ενός ή περισσοτέρων αρχείων και να εφαρμόζει μαζικά σε αυτά κάποιες ενέργειες. Οι ενέργειες αυτές θα μπορούσαν να είναι η αλλαγή/διόρθωση του ποσού της δημοπρασίας (δηλαδή να αλλάζει το ποσό αυτό σε όλα τα XML που υπάρχει το πρόβλημα), να μεταφέρει αρχεία από ένα σημείο του δικτύου σε κάποιο άλλο σημείο, να μπορεί να διαγράφει επιλεγμένες εγγραφές από την βάση, να μπορεί να κάνει overwrite αρχεία κα.

Δυσκολίες

Οι δυσκολίες δεν έλειψαν από κανένα κομμάτι της πτυχιακής μιας και όλες οι λειτουργίες απαιτούσαν προγραμματιστικές γνώσεις οι οποίες δεν έχουν καλυφθεί από την ύλη του εκπαιδευτικού προγράμματος (πλην του τρόπου

σύνδεσης της εφαρμογής με βάση). Παρ' αυτά όμως, οι γνώσεις ήταν αρκετές, έτσι ώστε με προσπάθεια, πολύ έρευνα και πολύ debugging (το δυσκολότερο κομμάτι ήταν το debugging των πολλαπλών threads) η εφαρμογή να λειτουργεί σωστά και αξιόπιστα, εδώ και αρκετά χρόνια, στην εταιρεία αποδελτίωσης εντύπων. Πέρα από τον προγραμματισμό, η εγκατάσταση του SQL Server, η σχεδίαση και η υλοποίηση της νέας βάσης ήταν μια μεγάλη πρόκληση.

Συμπέρασμα

Μέσα από την ανάπτυξη της εφαρμογής και την σχεδίαση της βάσης μου δόθηκε η ευκαιρία να δοκιμάσω τις δυνατότητές μου δίνοντας λύση σε πραγματικές συνθήκες και προβλήματα, να διευρύνω τις γνώσεις μου πάνω στον προγραμματισμό και τέλος να μελετήσω την Oracle και την MS SQL. Ελπίζω και εσείς να αποκομίσατε νέες γνώσεις και να απολαύσατε την διαδρομή.

Βιβλιογραφία

- Java Documentation (JDK 7)
http://download.oracle.com/javase/7/docs/api/index.html
- Microsoft SQL Server 2008 R2 Documentation
http://msdn.microsoft.com/en-us/library/bb418440%28v=SQL.10%29.aspx
- SQL Server Management Studio
http://msdn.microsoft.com/en-us/library/ms174173.aspx
- XML Format
http://www.w3schools.com/xml/default.asp
- XML Parsers
http://www.ibiblio.org/xml/books/xmljava/chapters/ch05s02.html#d0e7095
- Java Logging Overview
http://download.oracle.com/javase/1.4.2/docs/guide/util/logging/overview.html
- Nick Stephen' s blog
http://blogs.oracle.com/nickstephen/entry/java_redirecting_system_out_and
- Guarded Blocks
http://download.oracle.com/javase/tutorial/essential/concurrency/guardmeth.html
- SAX Tutorial
http://developerlife.com/tutorials/?p=29#50616
- Understanding anonymous authentication and the IUSR account
http://kb2.adobe.com/cps/153/tn_15378.html
- Implementing GRUD Operations Using Stored Procedures
http://www.databasejournal.com/features/mssql/article.php/3082201/Implementing-CRUD-Operations-Using-Stored-Procedures-Part-1.htm
- Cay S. Horstmann, Gary Cornell, 2004. Core Java 2 Volume II – Advanced Features. 7th ed. Prentice Hall PTR

ΠΑΡΑΡΤΗΜΑΤΑ

Παράρτημα Α

Δείγμα αρχείου XML – Άρθρο

```
<?xml version="1.0" encoding="UTF-16" ?>
- <clip xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <author>ΜΑΙΡΗ ΛΑΜΠΑΔΙΤΗ</author>
    <id>6084834</id>
    <artId>1433291</artId>
    <headline>Κοινωνικό πακέτο 300 εκατ. ευρώ υπέρ των ευπαθών ομάδων</headline>
    <pageName>4</pageName>
    <numberOfClippings>1</numberOfClippings>
    <articleSize>1065</articleSize>
    <articleType>1</articleType>
    <foreas />
    <pageSize>A3</pageSize>
    <publicationDate>2010-06-20</publicationDate>
    <timeCreated>2010-06-20T08:00:12</timeCreated>
    <searchWord>ΕΡΓΑΤΙΚΑ</searchWord>
    <searchWordId>1578</searchWordId>
    <publicationId>639</publicationId>
    <publicationName>ΠΡΩΤΟ ΘΕΜΑ</publicationName>
    <publicationCirc>144010</publicationCirc>
    <paper_type>Εφημερίδα / Κύρια@Πανελλαδικό</paper_type>
    <customerId>ΜΤΛ3</customerId>
    <customerName>ΒΙΟΧΑΛΚΟ</customerName>
    <cust_resp>Επίθετο Όνομα</cust_resp>
    <cust_city>ΑΘΗΝΑ</cust_city>
    <cust_email>sample@email.gr</cust_email>
    <agreementId>88</agreementId>
    <agreementName>ΑΡΘΡΑ</agreementName>
    <articlePDFFile>6084834.PDF</articlePDFFile>
    <articleText>Κοινωνικό πακέτο 300 εκατ. ευρώ υπέρ των ευπαθών ομάδων
     Σε ποίους θα μοιραστούν τα χρήματα που θα δοθούν ... [ ... παράληψη
     κειμένου ... ]</articleText>
  </clip>
```

Κώδικας 23: *Δείγμα αρχείου XML (άρθρο). Όνομα αρχείου 6084834.XML*

Δείγμα αρχείου XML - Δημοπρασία

```
<?xml version="1.0" encoding="UTF-16" ?>
- <clip xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <author />
    <id>6014937</id>
    <artId>1412154</artId>
    <headline>3810752/20-07-2010</headline>
    <pageName>36</pageName>
    <numberOfClippings>1</numberOfClippings>
    <articleSize>331</articleSize>
    <articleType>2</articleType>
    <foreas>ΟΑΕΔ</foreas>
    <pageSize>A3</pageSize>
    <publicationDate>2010-06-10</publicationDate>
    <timeCreated>2010-06-10T14:38:28</timeCreated>
    <searchWord>ΔΣΕC1</searchWord>
    <searchWordId>5578</searchWordId>
    <publicationCirc>1350</publicationCirc>
    <paper_type>Εφημερίδα / Κύρια@Πανελλαδικό</paper_type>
    <customerId>ΣΕΚ89</customerId>
    <customerName>ΑΦΟΙ ΝΙΚΟΛΟΥΔΗ</customerName>
    <cust_resp>Επίθετο Όνομα</cust_resp>
    <cust_city>ΜΕΤΑΜΟΡΦΩΣΗ ΑΤΤΙΚΗΣ</cust_city>
    <cust_email>sample@email2.com</cust_email>
    <agreementId>1001</agreementId>
    <agreementName>ΔΗΜΟΠΡΑΣΙΕΣ</agreementName>
    <articlePDFFile>6014937.PDF</articlePDFFile>
    <articleText>ΕΛΛΗΝΙΚΗ ΔΗΜΟΚΡΑΤΙΑ ΥΠΟΥΡΓΕΙΟ ΕΡΓΑΣΙΑ ΚΑΙ ΚΟΙΝΩΝΙΚΗΣ
      ΑΣΦΑΛΙΣΗΣ ΟΡΓΑΝΙΣΜΟΣ ΑΠΑΣΧΟΛΗΣΗΣ ΕΡΓΑΤΙΚΟΥ ΔΥΝΑΜΙΚΟΥ ΔΗΜΟΣΙΕΥΣΗ
      ΠΕΡΙΛΗΨΗΣ ΔΙΑΚΗΡΥΞΗΣ ιπτ' αριθμ.: 02/2010 Ανοικτού δημόσιου διεθνούς
      διαγωνισμού με σφραγισμένες προσφορές και κριτήριο κατακύρωσης την
      συμφερότερη προσφορά για την επιλογή Αναδόχου του έργου: «ΦΥΛΑΞΗ ΚΑΙ
      ΠΡΟΣΤΑΣΙΑ ΤΩΝ ΚΤΙΡΙΩΝ ΤΩΝ ΥΠΗΡΕΣΙΩΝ ΤΟΥ ΟΑΕΔ» 1. ΑΝΑΘΕΤΟΥΣΑ ΑΡΧΗ:
      Οργανισμός Απασχόλησης Εργατικού Δυναμικού (ΟΑΕΔ) ... [ ... παράληψη
      κειμένου ... ]</articleText>
  </clip>
```

Κώδικας 24: *Δείγμα αρχείου XML (δημοπρασία). Όνομα αρχείου 6014937.XML*

Ανάλυση σημάνσεων (tags)

Πίνακας 3: *Σημάνσεις που αφορούν το απόκομμα*

Clip	Η οντότητα του αποκόμματος
Author	Ο συντάκτης
Id	Ο μοναδικός κωδικός του αποκόμματος
ArtId	Ο μοναδικός κωδικός του άρθρου ή της δημοπρασίας
Headline	Ο τίτλος του άρθρου (για άρθρα) ή το ποσό με την ημερομηνία λήξης της δημοπρασίας (για δημοπρασίες)
PageName	Η σελίδα του άρθρου ή της δημοπρασίας
numberOfClippings	Ο συνολικός αριθμός των σελίδων που αποτελούν το απόκομμα
ArticleSize	Τα τετραγωνικά εκατοστά του αποκόμματος (όλων των σελίδων)
ArticleType	Ο τύπος του αποκόμματος, 1 για άρθρα, 2 για δημοπρασίες
Foreas	Ο φορέας της δημοπρασίας, κενό αν είναι άρθρο
PageSize	Το μέγεθος της σαρωμένης σελίδας
PublicationDate	Η ημερομηνία δημοσίευσης του εντύπου
TimeCreated	Ημερομηνία και ώρα δημιουργίας του XML εγγράφου διαχωρισμένα με τον χαρακτήρα 'T'
PublicationId	Ο μοναδικός κωδικός του εντύπου
PublicationName	Το όνομα του εντύπου
PublicationCirc	Η κυκλοφορία του εντύπου
paper_type	Ο τύπος του εντύπου
ArticlePDFFile	Το όνομα του συνοδευόμενου PDF αρχείου
ArticleText	Το OCR κείμενο

Πίνακας 4: *Σημάνσεις που αφορούν τον πελάτη*

SearchWord	Η γενική κατηγορία ενδιαφέροντος
SearchWordId	Ο κωδικός του seachWord
CustomerId	Ο κωδικός του πελάτη
CustomerName	Η επωνυμία του πελάτη
cust_resp	Ονοματεπώνυμο υπευθύνου
cust_city	Πόλη στην οποία βρίσκεται η εταιρία
cust_email	Το email του υπεύθυνου
AgreementId	Ο κωδικός του agreement
AgreementName	Το όνομα του agreement

Παράρτημα Β

DOCTYPE definition

```
<!– author: Evelina Rimpapova –>
<!DOCTYPE webServConn [
<!ELEMENT Configuration (root_paths, log_files, email_config, webServConn,
oracleServConn, convertCommand1, convertCommand2, threads,
oracle_mssql_synch )>

<!ELEMENT root_paths (general_input, redirectionPath,
pdf_output,output_onError,djvu_output,xml_output)>
<!ELEMENT general_input (#PCDATA) >
<!ATTLIST general_input interval CDATA "5">
<!ELEMENT redirectionPath (#PCDATA) >
<!ATTLIST redirectionPath ifFilesMoreThan  CDATA "200">
<!ELEMENT pdf_output (#PCDATA) >
<!ELEMENT output_onError (#PCDATA) >
<!ELEMENT djvu_output (#PCDATA) >
<!ELEMENT xml_output (#PCDATA) >

<!ELEMENT log_files (log+)>
<!ELEMENT log (#PCDATA)>
<!ATTLIST log maxSizeInMB CDATA "1">
<!ATTLIST log numberOfLogFiles CDATA "1">
<!ATTLIST log append (true|false) "true">

<!ELEMENT email_config (email+)>
<!ELEMENT email (host,from,to*,subject,(body?|body_sample_file?))>
<!ELEMENT host (#PCDATA) >
<!ELEMENT from (#PCDATA) >
<!ELEMENT to (#PCDATA) >
<!ELEMENT subject (#PCDATA) >
<!ELEMENT body (#PCDATA) >
<!ELEMENT body_sample_file (#PCDATA) >
```

Κώδικας 25: *Ορισμός Doctype (τμήμα 1)*

```
<!ELEMENT webServConn (IP, databaseName, username, password)>
<!ELEMENT IP (#PCDATA) >
<!ELEMENT databaseName (#PCDATA) >
<!ELEMENT username (#PCDATA) >
<!ELEMENT password (#PCDATA) >

<!ELEMENT oracleServConn (OracleIP, OracledatabaseName, Oracleusername,
Oraclepassword, OracleInternalLogon)>
<!ELEMENT OracleIP (#PCDATA) >
<!ELEMENT OracledatabaseName (#PCDATA) >
<!ELEMENT Oracleusername (#PCDATA) >
<!ELEMENT Oraclepassword (#PCDATA) >
<!ELEMENT OracleInternalLogon (#PCDATA) >

<!ELEMENT convertCommand1 (command, profile)>
<!ELEMENT command (#PCDATA) >
<!ELEMENT profile (#PCDATA) >

<!ELEMENT convertCommand2 (programPath, options)>
<!ELEMENT programPath (#PCDATA) >
<!ELEMENT options (#PCDATA) >

<!ELEMENT threads (number)>
<!ELEMENT number (#PCDATA) >

<!ELEMENT oracle_mssql_synch (enabled, start_time, interval)>
<!ELEMENT enabled (#PCDATA)  >
<!ELEMENT start_time (#PCDATA) >
<!ELEMENT interval (#PCDATA) >
]>
```

Κώδικας 26: *Ορισμός Doctype (τμήμα 2)*

Δείγμα αρχείου Configuration.xml

```
<!-- author: Evelina Rimpapova -->
- <Configuration>
  - <root_paths>
      <general_input interval="5">C:\local_folder\input</general_input>
      <!-- Διεύθυνση σε τοπικό δίσκο -->
      <redirectionPath ifFilesMoreThan="5">\\Server\local_folder_input</redirectionPath>
      <pdf_output>\\Server2\local_folder\PdfRootPath</pdf_output>
      <output_onError>C:\local_folder\onError</output_onError>
      <djvu_output>\\Server2\local_folder\DjVuRootPath</djvu_output>
      <xml_output>\\Server2\local_folder\XmlRootPath</xml_output>
    </root_paths>
  - <log_files>
      <log maxSizeInMB="1" numberOfLogFiles="1" append="true">C:\logPart1.txt</log>
      <log maxSizeInMB="5" numberOfLogFiles="2" append="true">C:\logPart2.txt</log>
      <log maxSizeInMB="3" numberOfLogFiles="1" append="true">C:\Synchronization_log.txt</log>
    </log_files>
  - <email_config>
    - <email>
        <host>smtp.mail.server.gr</host>
        <from>WebApplication@companyDomain.gr</from>
        <to>support_email@companyDomain.gr</to>
        <subject>Error during convertion of pdf file.</subject>
      - <body_sample_file>
          C:\Program Files\Convert Application\TextFiles\ErrorConvertingMail.txt
        </body_sample_file>
      </email>
    - <email>
        <host>smtp.mail.server.gr</host>
        <from>info@companyDomain.gr</from>
        <to>info@companyDomain.gr</to>
        <!-- πρώτη κοινοποίηση σε αυτό το email -->
      - <subject>
          Clip News - Αποστολή κωδικών αποδελτίωσης για την υπηρεσία Web εφαρμογής
        </subject>
      - <body_sample_file>
          C:\Program Files\Convert Application\TextFiles\newClient_emailSample.txt
        </body_sample_file>
      </email>
```

Κώδικας 27: *Δείγμα αρχείου Configuration.xml (τμήμα 1)*

```
- <email>
    <host>smtp.mail.server.gr</host>
    <from>WebApplication@ClipNews.gr</from>
    <to>support_email@companyDomain.gr</to>
    <subject>Error during xml insertion.</subject>
  - <body>
      Could't insert the following xml file in the database:
    </body>
  </email>
- <email>
    <host>smtp.mail.server.gr</host>
    <from>WebApplication@ClipNews.gr</from>
    <to>support_email@companyDomain.gr</to>
    <subject>Error during xml insertion.</subject>
  - <body>
      Pdf found without the following xml file. For that reason the pdf file was moved in the error folder.
    </body>
  </email>
</email_config>
- <webServConn>
    <IP>Server2_IP</IP>
    <databaseName>portal_database_name</databaseName>
    <username>A_Yser_n@me</username>
    <password>a_P@ssword_</password>
</webServConn>
- <oracleServConn>
    <OracleIP>Server3:Port</OracleIP>
    <OracledatabaseName>prodaction_database_name</OracledatabaseName>
    <Oracleusername>A_Yser_n@me2</Oracleusername>
    <Oraclepassword>a_P@ssword_2</Oraclepassword>
    <OracleInternalLogon>sys</OracleInternalLogon>
</oracleServConn>
- <convertCommand1>
    <command>documenttodjvu.exe</command>
    <profile>--profile=convert</profile>
</convertCommand1>
- <convertCommand2>
    <programPath>C:\Program Files\Pdf2Djvu\pdf2djvu.exe</programPath>
    <options>-d 300 --bg-subsample=1 --media-box</options>
</convertCommand2>
```

Κώδικας 28: *Δείγμα αρχείου Configuration.xml (τμήμα 2)*

```
- <threads>
    <number>6</number>
  </threads>
- <oracle_mssql_synch>
    <enabled>true</enabled>
    <start_time>22</start_time>
    <!-- HH (24h)-->
    <interval>24</interval>
    <!-- in hours -->
  </oracle_mssql_synch>
</Configuration>
```

Κώδικας 29: *Δείγμα αρχείου Configuration.xml (τμήμα 3)*

Επεξήγηση σημάνσεων αρχείου παραμετροποίησης

Πίνακας 5: *Επεξήγηση σημάνσεων (tags)*

General_Input	Η διεύθυνση του φακέλου από τον οποίο η εφαρμογή θα ελέγχει ανά 'Χ' χρονικό διάστημα (το οποίο ορίζεται από το attribute `general_input interval`) για αρχεία PDF και XML, τα οποία θα έχουν αποθηκευτεί στην διεύθυνση αυτή από το παραγωγικό υποσύστημα.
RedirectionPath	Η διεύθυνση του φακέλου όπου θα στέλνονται τα αρχεία PDF και XML στην περίπτωση που ο αριθμός των αρχείων (δυάδων) στο General_input ξεπερνάει την τιμή του attribute `ifFilesMoreThan` (και μόνο για όσο την ξεπερνά).
PDF_output	Η διεύθυνση όπου θα αποθηκεύονται τα PDF και θα είναι πλέον προσβάσιμα από τους πελάτες του Web portal.
Djvu_output	Η διεύθυνση όπου θα αποθηκεύονται τα Djvu και θα είναι πλέον προσβάσιμα από τους πελάτες του Web portal.
XML_output	Η διεύθυνση όπου θα αποθηκεύονται τα XML.
output_onError	Η διεύθυνση όπου αποθηκεύονται τα αρχεία τα οποία για τον οποιονδήποτε λόγο δεν μπόρεσαν να επεξεργαστούν από την εφαρμογή.
Log	Η διεύθυνση όπου αποθηκεύονται τα log files της εφαρμογής. Το κάθε ένα από αυτά τα αρχεία δεν μπορεί να ξεπεράσει το μέγεθος το οποίο ορίζεται από το attribute maxSizeInMB. Όταν όμως ορίζουμε μεγάλο μέγιστο μέγεθος log αρχείου (πχ 10MB) αυτό είναι δύσκολα διαχειρίσιμο λόγο του μεγέθους του (κάνει πολύ ώρα να ανοίξει με ένα Notepad). Σε αυτή την περίπτωση είναι προτιμότερο να έχουμε 5 log αρχεία με μέγιστο μέγεθος τα 2MB το καθένα. Έτσι, μέσω του attribute mumberOfLogFiles μπορούμε να ορίσουμε το

	σε πόσα αρχεία θα καταμερίζονται οι εγγραφές του logging της εφαρμογής. Το attribute append ορίζει αν θέλουμε να γίνεται overwrite ή append η κάθε νέα εγγραφή του logging.
Email	Περιλαμβάνει τις παραμέτρους έτσι ώστε η εφαρμογή να στέλνει email. Περιλαμβάνονται τα attributes host, from, to, subject και body τα οποία είναι απαραίτητα για να μπορεί να πραγματοποιηθεί η αποστολή του email. Τέλος, το body_sample_file είναι η διεύθυνση ενός text αρχείου όπου το περιεχόμενο του οποίου θα τοποθετηθεί ως body του email. Η επιλογή αυτή μπορεί να χρησιμοποιηθεί ως εναλλακτική της επιλογής body.
IP	Η IP διεύθυνση του Server στον οποίο φιλοξενείτε το DBMS του υποσυστήματος δημοσιοποίησης.
databaseName	Το όνομα της βάσης του υποσυστήματος δημοσιοποίησης.
Username	Το απαραίτητο για την σύνδεση της εφαρμογής με την βάση του υποσυστήματος δημοσίευσης username.
Password	Το απαραίτητο για την σύνδεση της εφαρμογής με την βάση του υποσυστήματος δημοσίευσης password.
OracleIP	Η IP διεύθυνση του Server στον οποίο φιλοξενείται το DBMS του παραγωγικού υποσυστήματος.
OracledatabaseName	Το όνομα της βάσης του παραγωγικού υποσυστήματος.
Oracleusername	Το απαραίτητο για την σύνδεση της εφαρμογής με την βάση του παραγωγικού υποσυστήματος username.
Oraclepassword	Το απαραίτητο για την σύνδεση της εφαρμογής με την βάση του παραγωγικού υποσυστήματος password.
OracleInternalLogon	Το απαραίτητο (ειδικά για Oracle) για την σύνδεση της εφαρμογής με την βάση του παραγωγικού υποσυστήματος. Με αυτή την μεταβλητή ορίζονται τα δικαιώματα του χρήστη που κάνει login (πχ login ως SYS ή SYSDBA)
Command	Η εντολή (command line) η οποία πρέπει να εκτελεστεί

	προκειμένου η εφαρμογή να μπορέσει να «καλέσει» το εξωτερικό πρόγραμμα το οποίο εκτελεί την μετατροπή αρχείου PDF σε αρχείο DjVu. Όπως αναφέραμε, η μετατροπή των αρχείων γίνεται από δύο εξωτερικά προγράμματα. Αυτή είναι η εντολή για την χρήση του πρώτου προγράμματος, που είναι εμπορικό.
Profile	Το όνομα του profile το οποίο έχει δημιουργηθεί και περιέχει τις παραμέτρους μετατροπής και είναι απαραίτητο στην πρώτη εφαρμογή για την μετατροπή αρχείου PDF σε DjVu.
ProgramPath	Η διεύθυνση όπου έχει εγκατασταθεί το πρόγραμμα το οποίο θα πραγματοποιήσει την μετατροπή των αρχείων PDF σε DjVu με βάση. Το πρόγραμμα αυτό θα καλείται μόνο στην περίπτωση που το πρώτο πρόγραμμα δεν μπόρεσε να πραγματοποιήσει την μετατροπή.
Options	Οι παράμετροι απαραίτητες για την μετατροπή (για το δεύτερο πρόγραμμα)
Threads	Ο αριθμός των threads που θέλουμε να τρέχει η εφαρμογή.
Enabled	Αν θέλουμε η εφαρμογή να κάνει συγχρονισμό των δύο βάσεων θα πρέπει η τιμή αυτού του element να είναι true. Διαφορετικά θα πρέπει να είναι false. Όταν έχουν εγκατασταθεί πάνω από μία εφαρμογές, είναι καλύτερα μόνο μία να πραγματοποιεί τον συγχρονισμό.
Start_time	Η ώρα κατά την οποία θα ξεκινήσει ο συγχρονισμός.
Interval	Αριθμός, ο οποίος δηλώνει ανά πόσες ώρες θέλουμε να γίνεται ο συγχρονισμός.

Παράρτημα Γ

Εγκατάσταση MS SQL Server

Για να μπορέσουμε να στήσουμε την βάση μας, πρώτο βήμα αποτελεί η εγκατάσταση του MS SQL Server 2008 R2 το οποίο αποτελεί την τρέχουσα έκδοση, χωρίς περιορισμούς, με όλα τα πιθανά features και μπορείτε να το εγκαταστήσετε ως evaluation για την χρονική περίοδο των έξι μηνών. Υπάρχει και η Express έκδοση η οποία είναι δωρεάν αλλά έχει κάποιους περιορισμούς ως προς το μέγεθος της βάσης (δεν μπορεί να είναι πάνω από 2GB), δεν μπορείς να έχεις πάνω από ένα instance και λείπουν πολλά advanced features. Τα αρχεία εγκατάστασης μπορείτε να τα βρείτε στην ακόλουθη ιστοσελίδα της Microsoft (http://www.microsoft.com/sqlserver/en/us/default.aspx).

Πριν την εγκατάσταση ίσως χρειαστείτε κάποια fixes της Microsoft, τον Windows Installer 4.5, και το νεότερο **.**NET Framework 3.5. Αν δεν τα έχετε εγκατεστημένα, ξεκινώντας την εγκατάσταση του SQL Server, θα ζητήσει την εγκατάστασή τους. Ακολουθούν τα βήματα της εγκατάστασης.

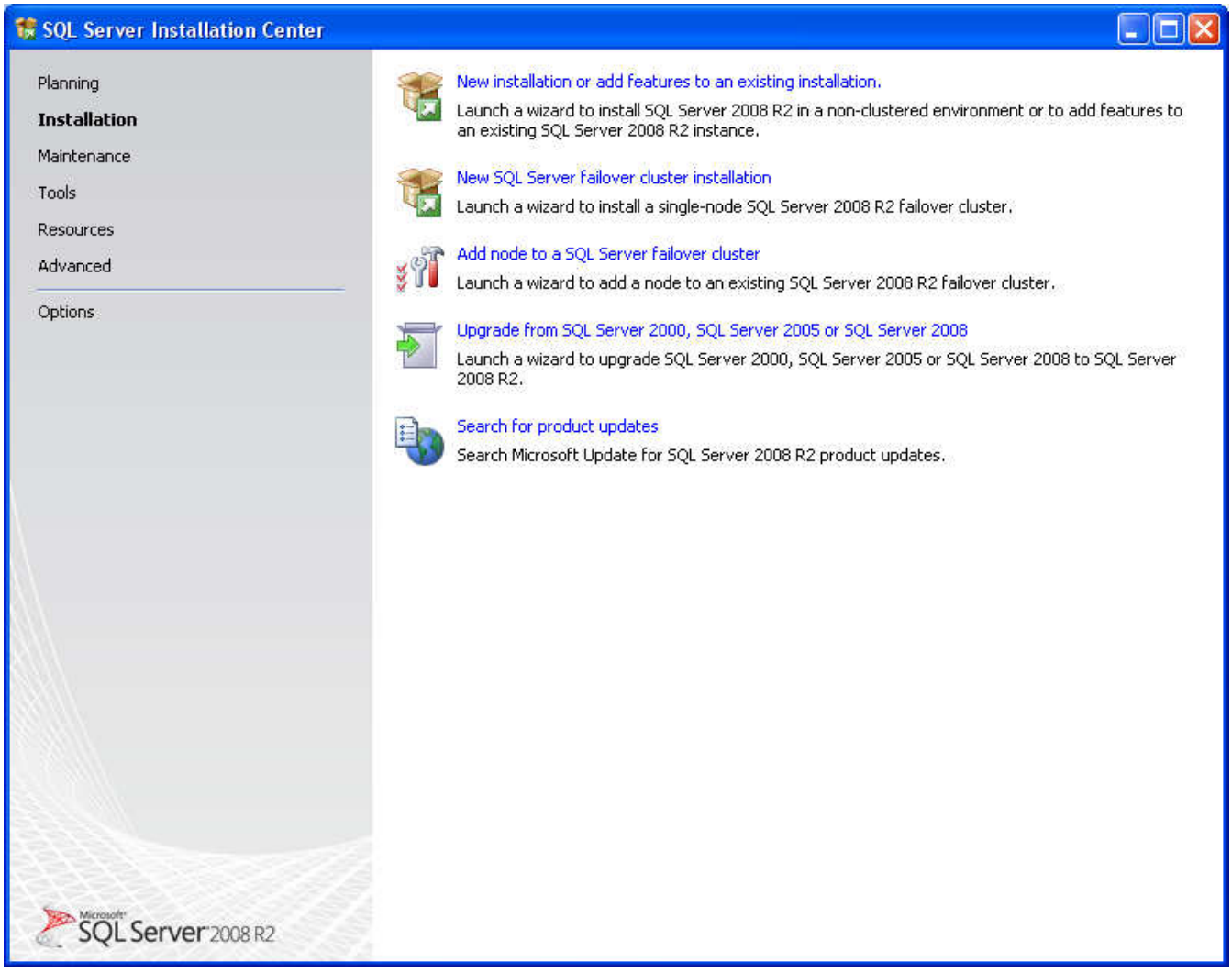
Βήμα 1.

Ανοίγοντας το αρχείο εγκατάστασης εμφανίζεται η ακόλουθη εικόνα με πολλές χρήσιμες πληροφορίες. Οι επιλογές που θα κάνετε στα επόμενα βήματα θα επηρεάσουν και τον αριθμό των βημάτων που θα χρειαστεί να κάνετε για την εγκατάσταση. Είναι πολύ πιθανό, κάποιο από τα ακόλουθα παράθυρα να μην εμφανιστεί κατά την διάρκεια της εγκατάστασης.

Εικόνα 24: *Εγκατάσταση MSSQL Server - Planning*

Βήμα 2.

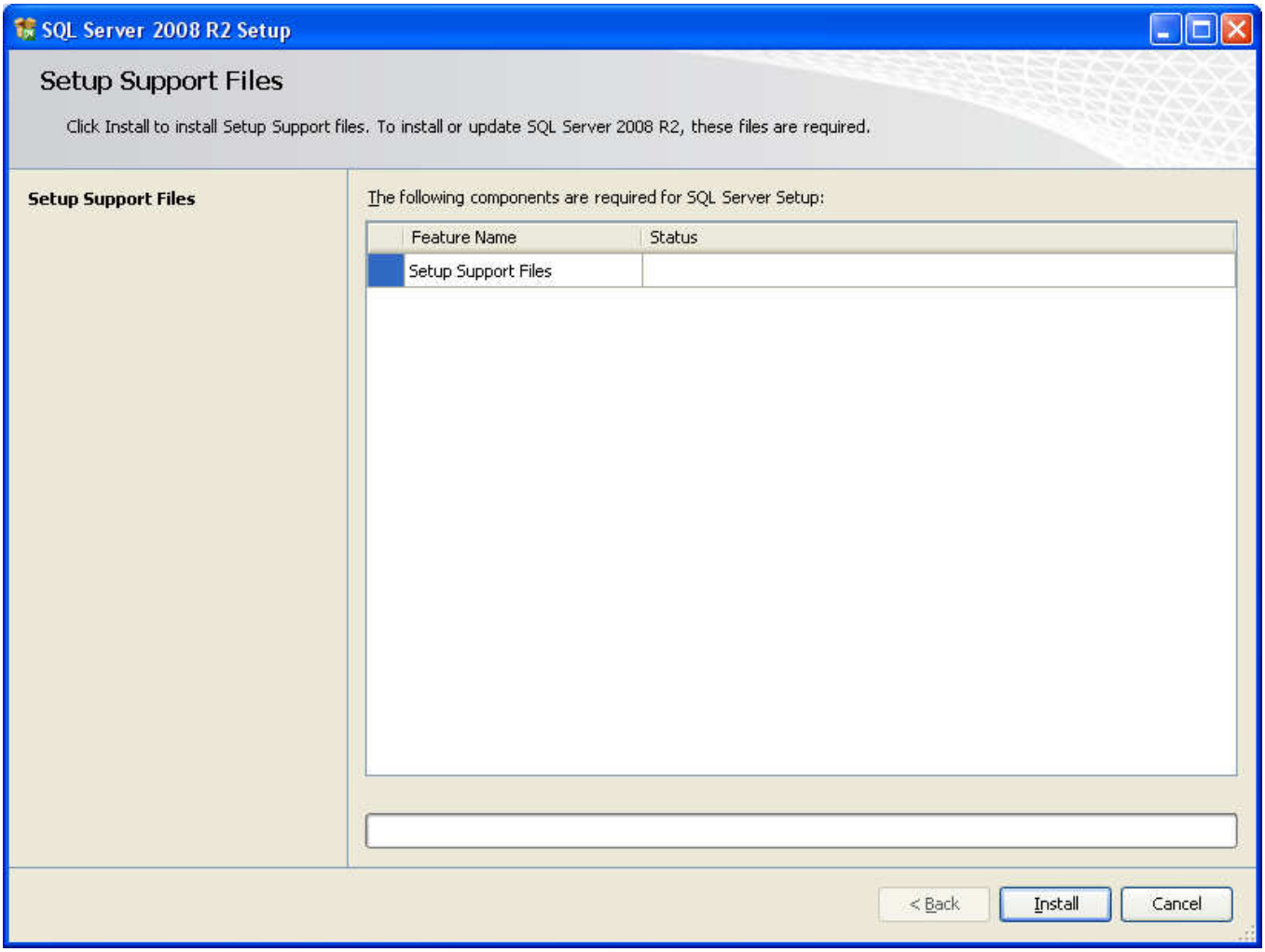
Για να προχωρήσουμε στην εγκατάσταση επιλέγουμε το “Installation” από το αριστερό μενού. Στη συνέχεια επιλέγουμε “New installation or add features to an existing installation”. Αυτόματα γίνεται έλεγχος του υπολογιστή για τυχόν προβλήματα τα οποία μπορεί να προκύψουν αν υπάρχει κάποια ασυμβατότητα. Αυτό θα επαναληφθεί και άλλες φορές κατά την παραμετροποίηση της εγκατάστασης πριν την τελική ευθεία.



Εικόνα 25: *Εγκατάσταση MSSQL Server - Installation*

Βήμα 3.

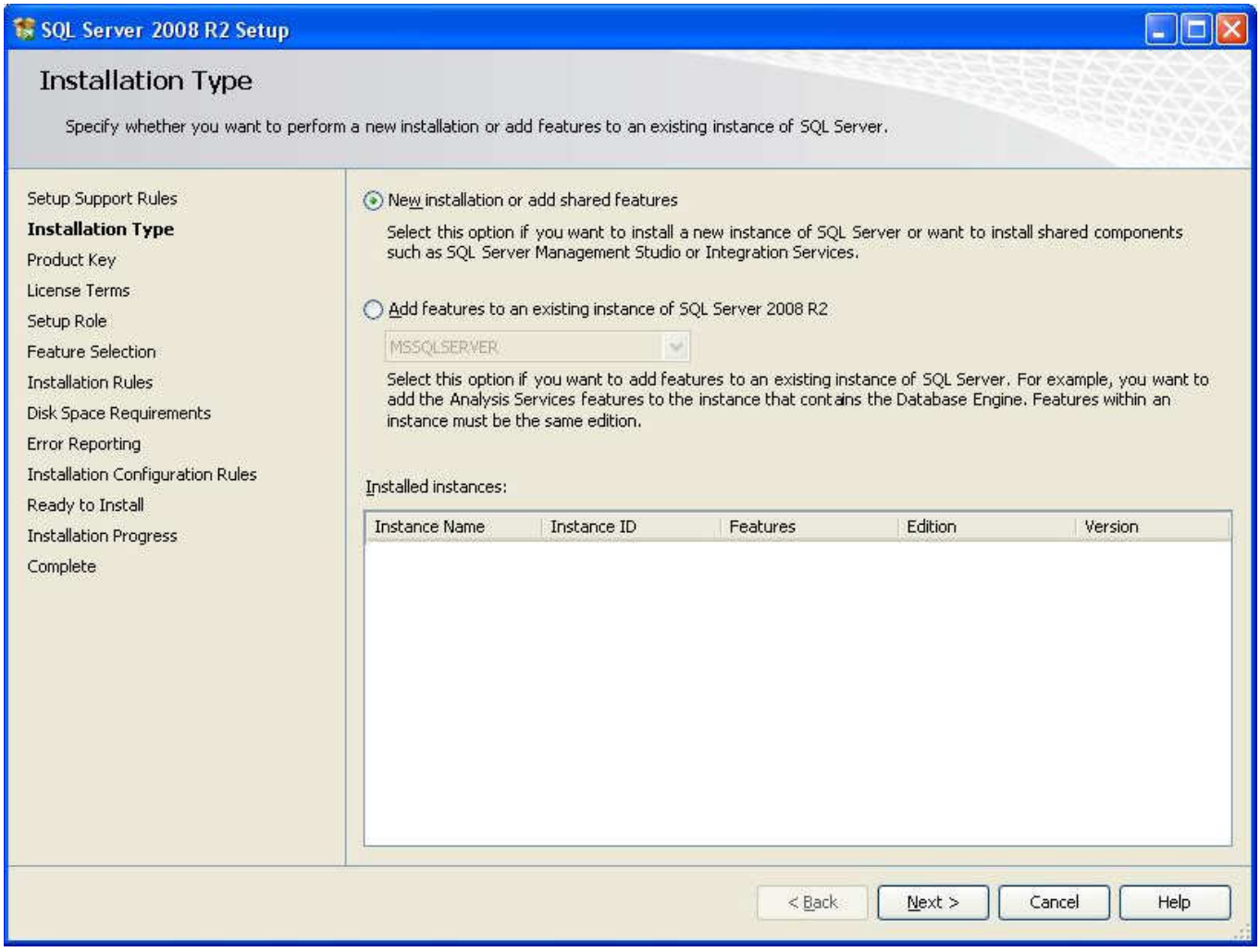
Αν και μόνο αν όλα πήγαν καλά κατά τον έλεγχο, προχωράμε στο επόμενο βήμα και κάνουμε click στο Install.



Εικόνα 26: *Εγκατάσταση MSSQL Server – Setup Support Files*

Βήμα 4.

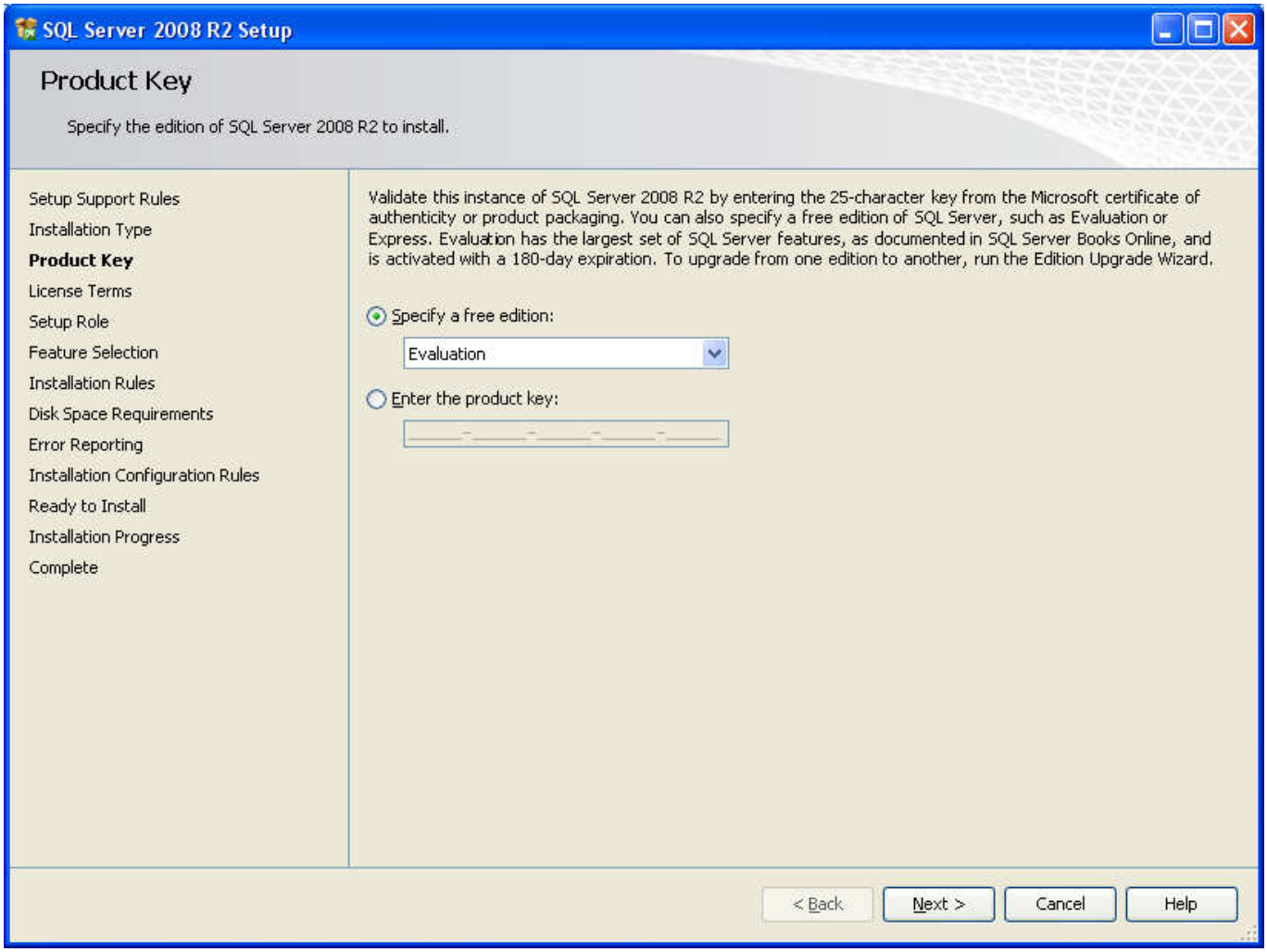
Στη συνέχεια πρέπει να επιλέξουμε το “New installation or add shared features” για νέα εγκατάσταση εκτός και αν θέλουμε να προσθέσουμε νέες δυνατότητες σε είδη εγκατεστημένο instance του SQL Server.



Εικόνα 27: *Εγκατάσταση MSSQL Server - Installation Type*

Βήμα 5.

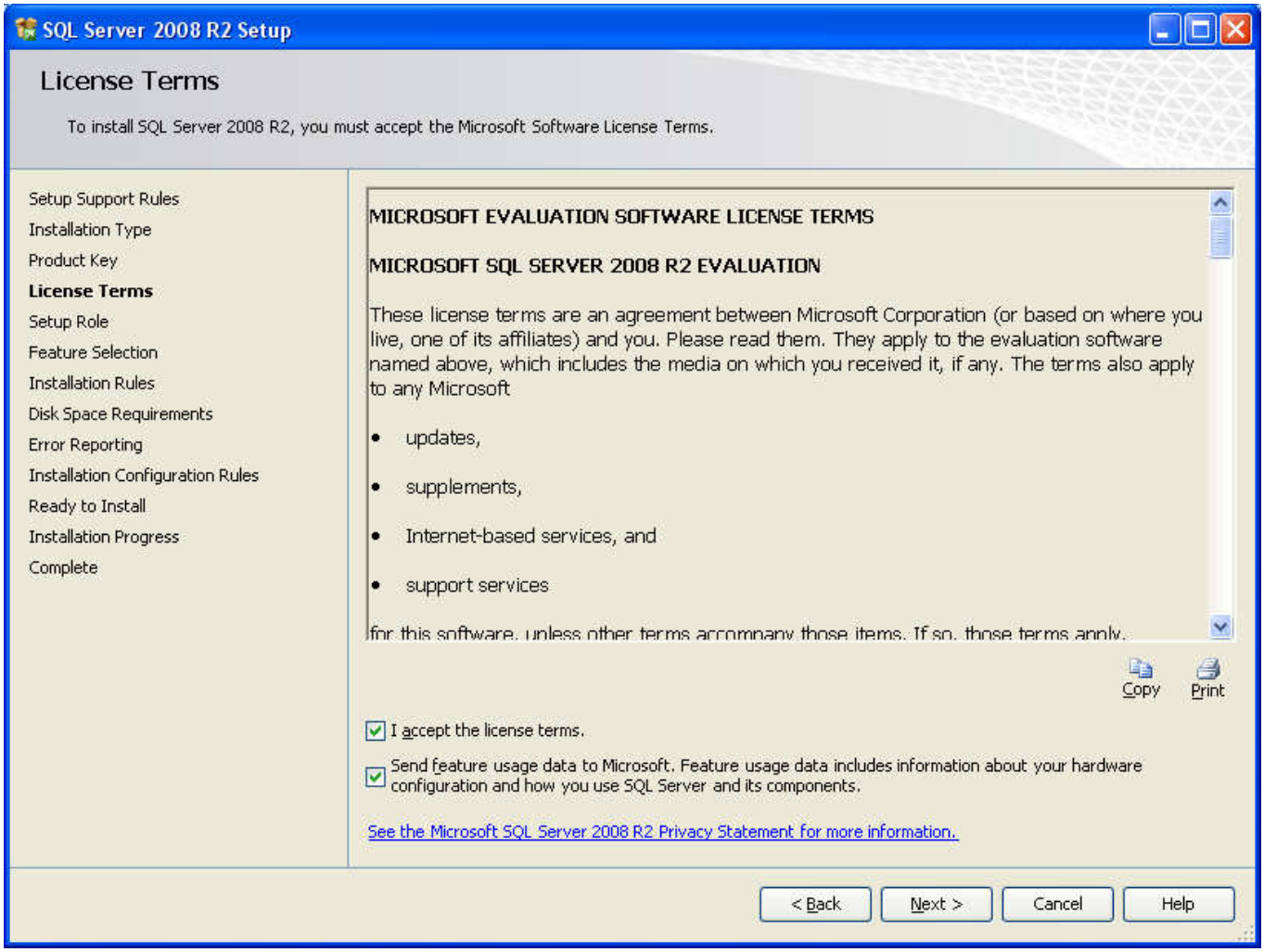
Επιλέγουμε Next.



Εικόνα 28: *Εγκατάσταση MSSQL Server - Product Key*

Βήμα 6.

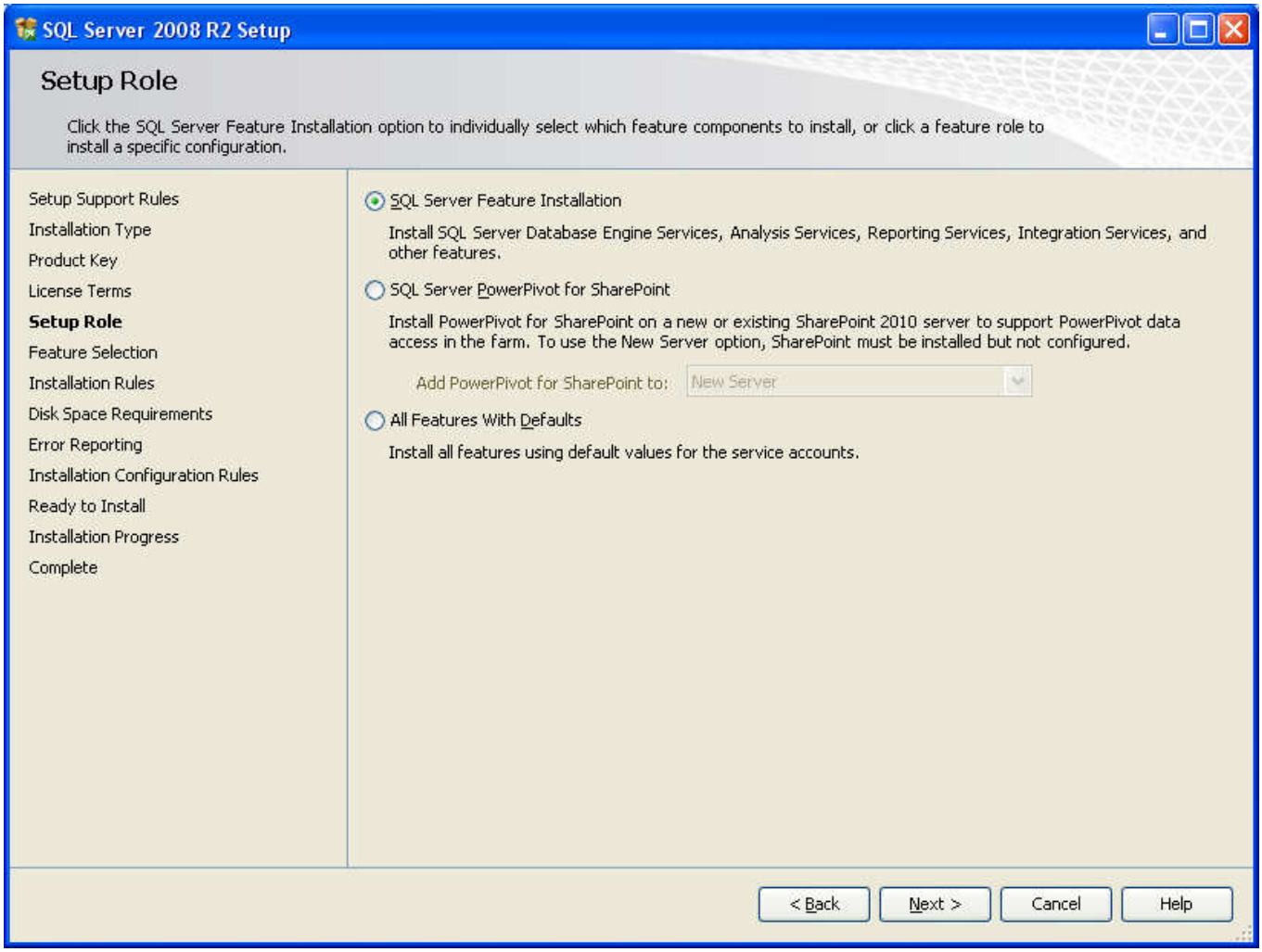
Επιλέγουμε τα δύο check boxes και επιλέγουμε Next.



Εικόνα 29: *Εγκατάσταση MSSQL Server - License terms*

Βήμα 7.

Επιλέγουμε SQL Server Feature Installation.



Εικόνα 30: *Εγκατάσταση MSSQL Server - Setup Role*

Βήμα 8.

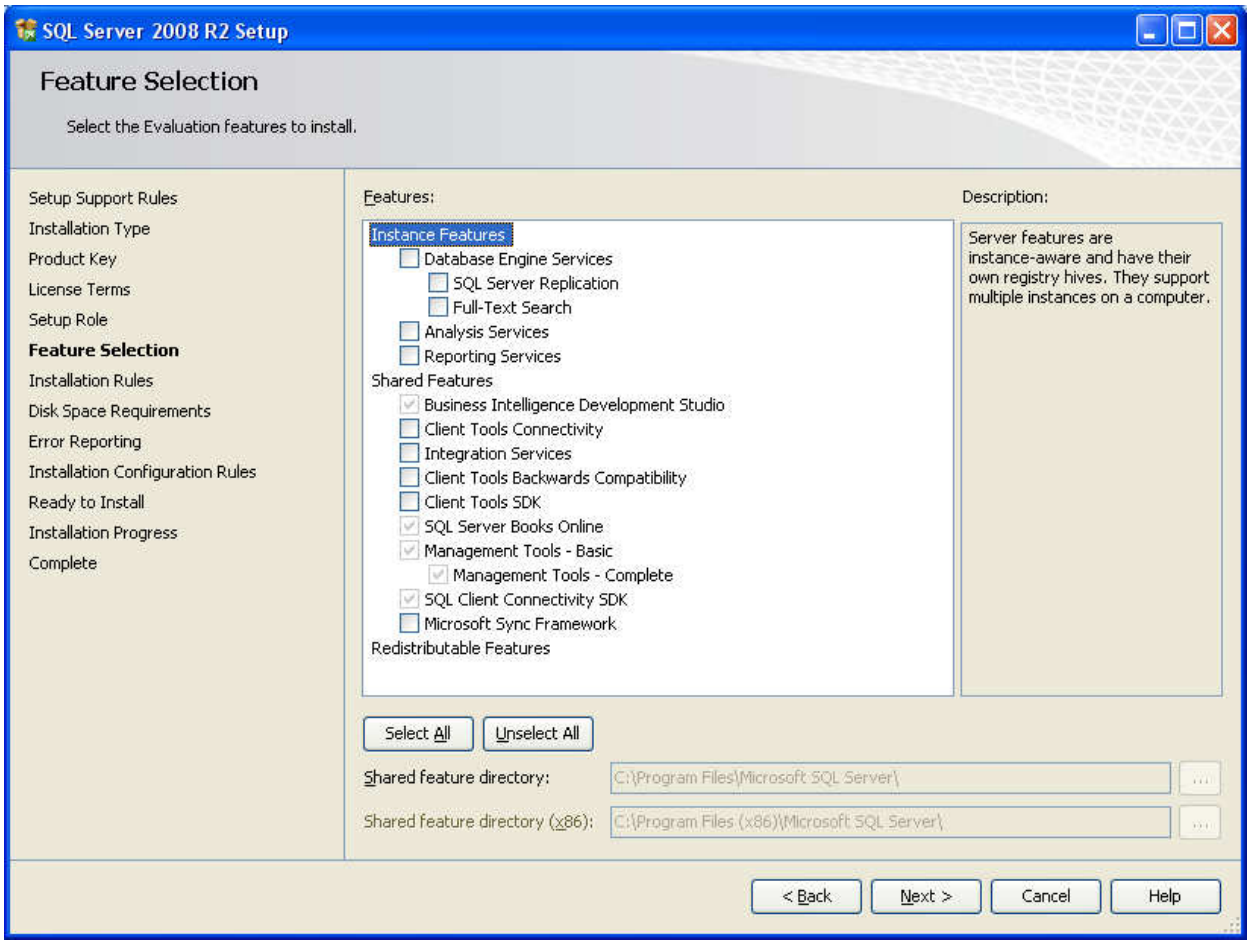
Για την εγκατάσταση του Service για βάσεις δεδομένων πρέπει να επιλέξουμε το “Database Engine Services”. Επιπρόσθετα πρέπει απαραίτητα να εγκαταστήσουμε και το Management Tools – Complete έτσι ώστε να γίνει και η εγκατάσταση του SQL Management Studio. Αυτό είναι το βασικό πρόγραμμα διαχείρισης το οποίο μας επιτρέπει να αλληλεπιδρούμε με την βάση, δηλαδή να δημιουργούμε νέες βάσεις, πίνακες, να τρέχουμε query κλπ. Δεν χρειάζεται να επιλέξουμε κάτι άλλο αλλά ας ρίξουμε μια ματιά στις σημαντικές δυνατότητες ενός Instance.

SQL Server Replication – Αυτό το Service δίνει την δυνατότητα παράλληλης λειτουργίας αντίγραφων της φάσης και συγχρονισμού αυτών.

Full-Text Search – Προσθέτει την δυνατότητα γρηγορότερης αναζήτησης (indexing) σε πεδία κειμένου μιας βάσης.

Analysis Services – Δίνει την δυνατότητα εφαρμογής Data Mining

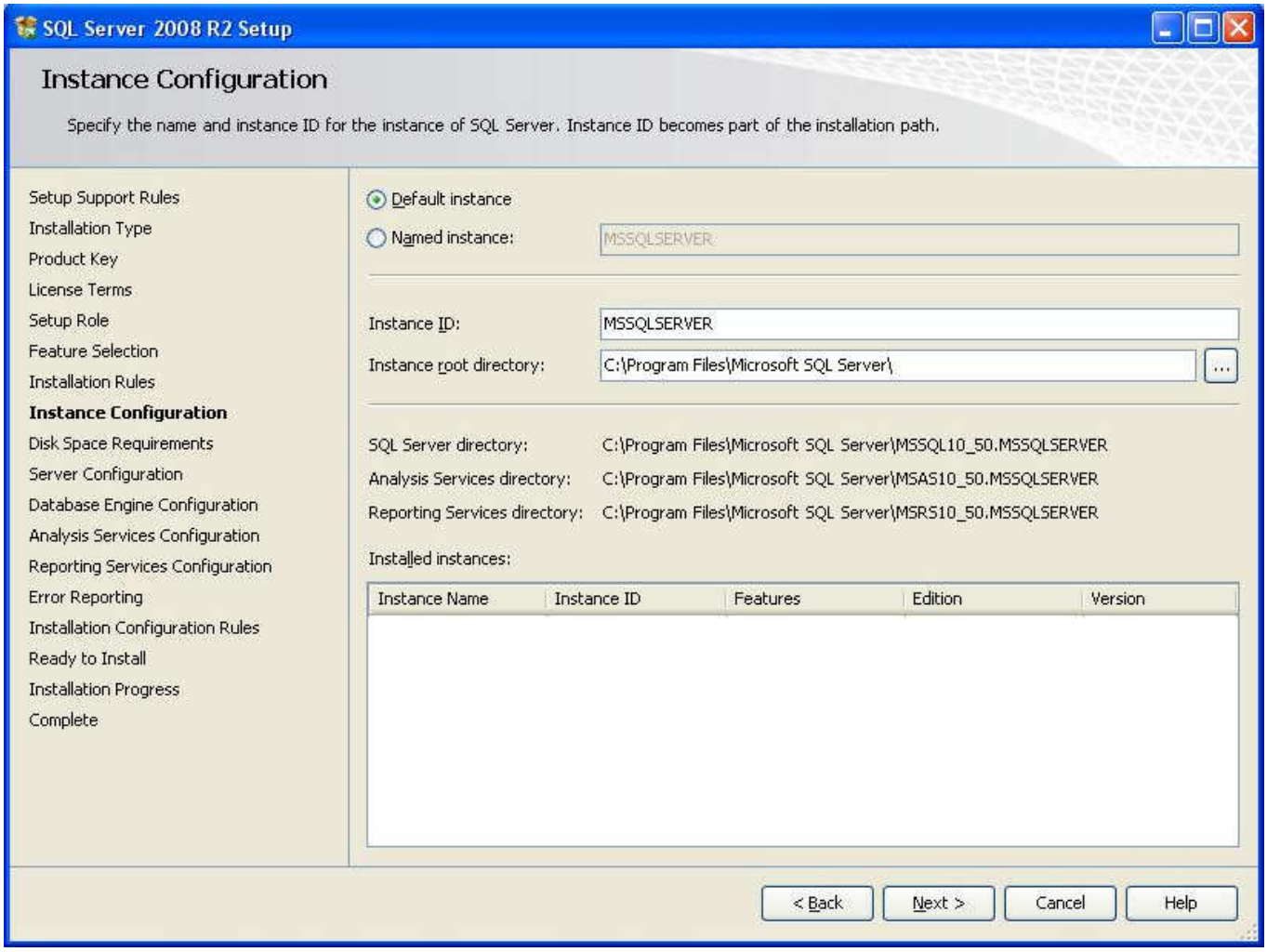
Reporting Services – Δημιουργία γραφημάτων και αναφορών που προκύπτουν από καθορισμένα από εμάς δεδομένα της βάσης.



Εικόνα 31: *Εγκατάσταση MSSQL Server - Feature Selection*

Βήμα 9.

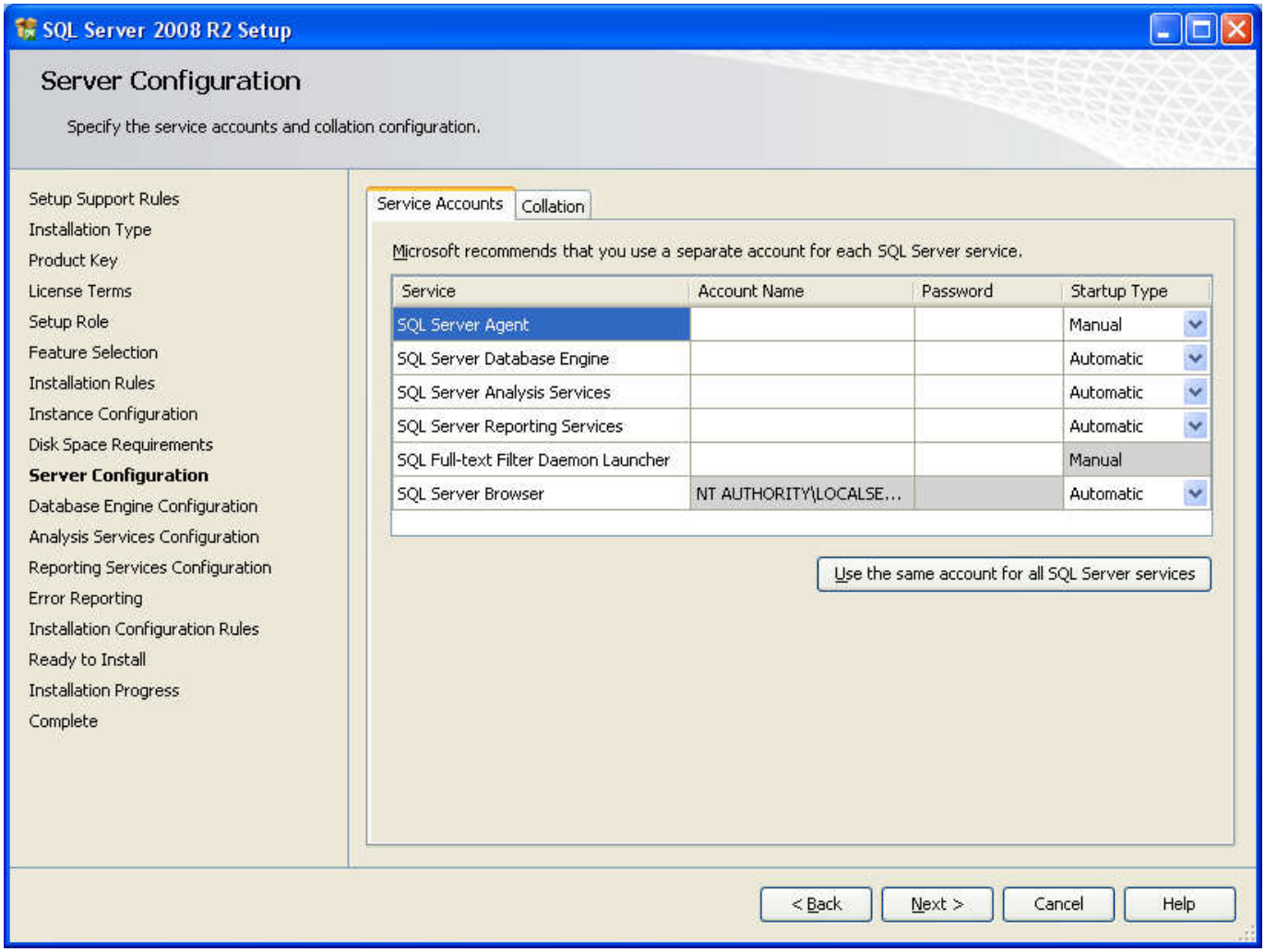
Στην νέα καρτέλα καλούμαστε να ονοματίσουμε το νέο instance ή να δεχτούμε το Default. Το όνομα που θα δώσουμε θα είναι το όνομα του Service το οποίο θα διαχειρίζεται τις βάσεις



Εικόνα 32: *Εγκατάσταση MSSQL Server - Instance Configuration*

Βήμα 10.

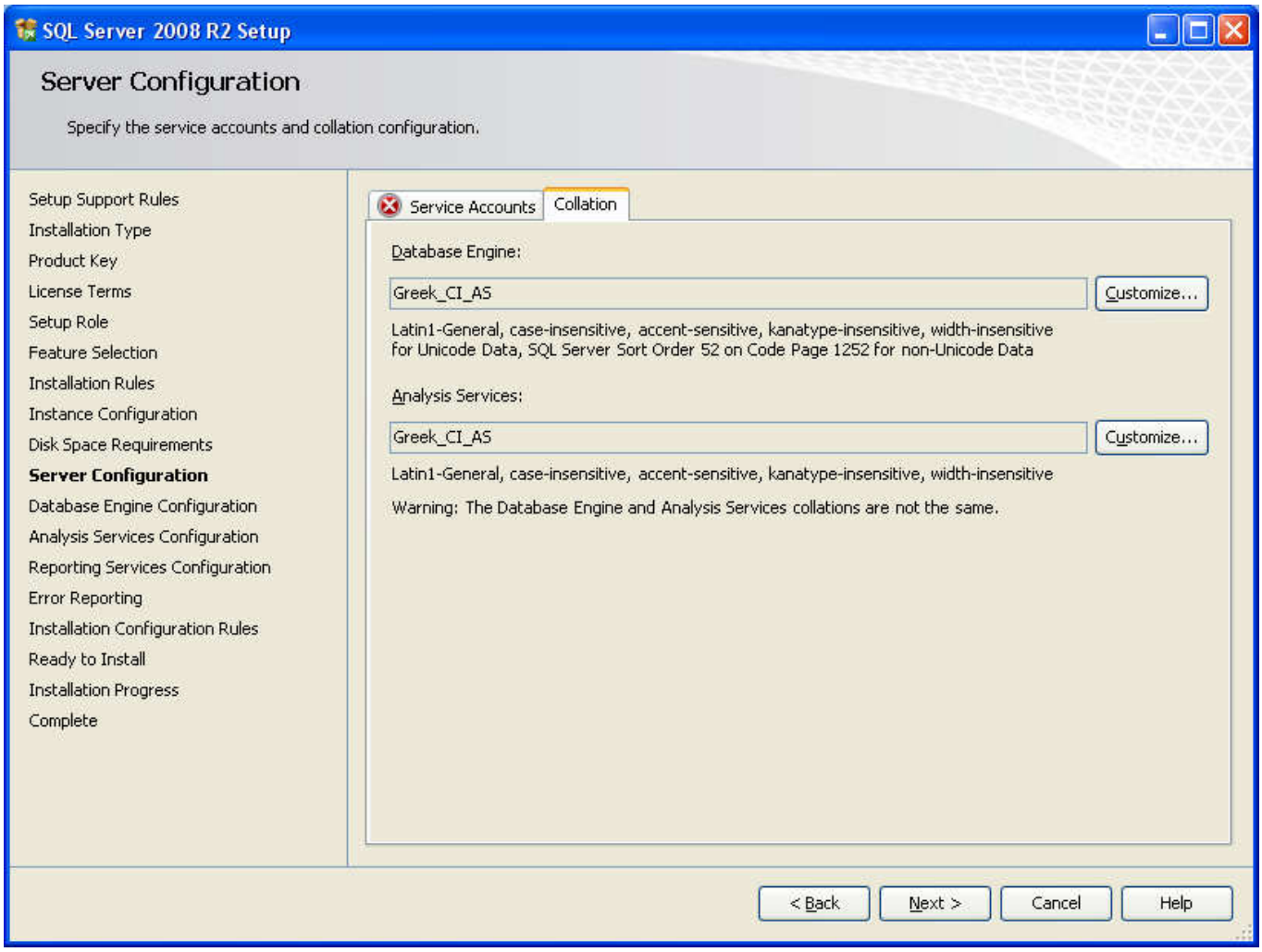
Στο επόμενο παράθυρο πρέπει να καθορίσουμε με ποιόν λογαριασμό θα τρέχει το κάθε επιμέρους Service του Instance. Ο λογαριασμός αυτός πρέπει να έχει δικαιώματα διαχειριστή και να μην έχει κενό password. Μπορούμε να βάλουμε τον ίδιο λογαριασμό για όλα τα services ή να επιλέξουμε διαφορετικό για το καθένα. Τα εμφανιζόμενα services μπορεί να διαφέρουν ανάλογα με το τι υπηρεσίες επιλέξατε να εγκαταστήσετε στο βήμα 8.



Εικόνα 33: *Εγκατάσταση MSSQL Server - Server Configuration – Service Accounts*

Βήμα 11.

Στην δεύτερη καρτέλα επιλέξτε το Collation. Το Collation ελέγχει την κωδικοσελίδα την οποία θα χρησιμοποιεί η βάση στην περίπτωση μη Unicode χαρακτήρων όπως και τον τρόπο που τα δεδομένα αυτά θα ταξινομούνται. Η επιλογή που εμφανίζεται από Default είναι αυτή που είναι ρυθμισμένη στον υπολογιστή/server όπου γίνεται η εγκατάσταση. Στην πιο κάτω περίπτωση έχουμε την ελληνική κωδικοσελίδα όπου δεν γίνεται διάκριση πεζών-κεφαλαίων (CI specifies case-insensitive) και δεν γίνεται διάκριση των τονισμένων χαρακτήρων από τους μη τονισμένους (AI specifies accent-insensitive).

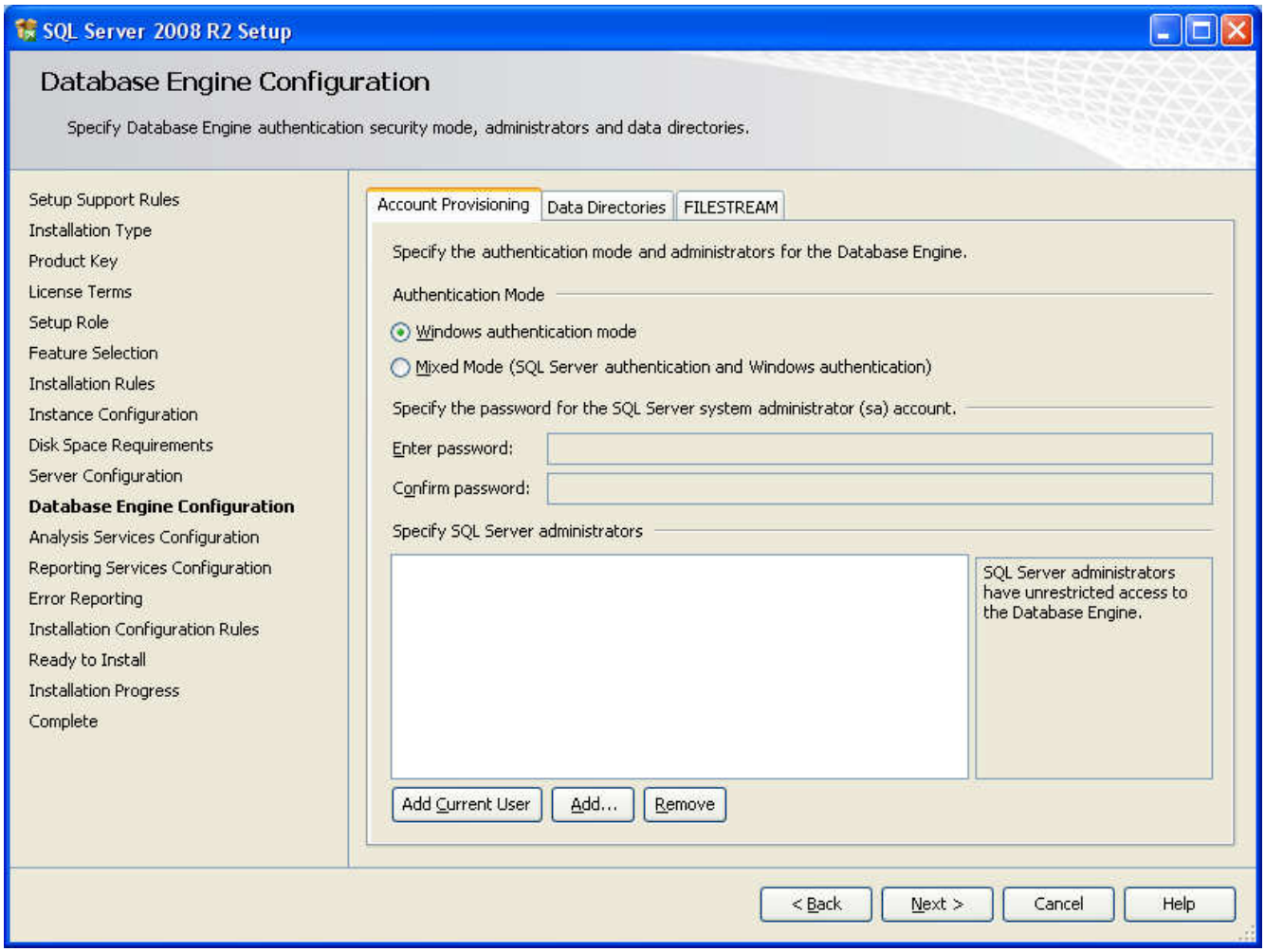


Εικόνα 34: *Εγκατάσταση MSSQL Server - Server Configuration – Collation*

Βήμα 12.

Στο επόμενο παράθυρο πρέπει να κάνετε την επιλογή για το πώς θα γίνεται η επικύρωση εισόδου (login) για την σύνδεση των χρηστών (clients) με τη βάση. Η πρώτη επιλογή είναι η Windows authentication mode, που σημαίνει πως ο λογαριασμός που ζητήσει Login θα πρέπει απαραίτητα να αναγνωρίζεται από τα Windows του μηχανήματος που κάνετε την εγκατάσταση. Αν επιλέξετε Mixed Mode έχετε την επιπλέον δυνατότητα να επιλέξετε και SQL authentication κατά το login, δηλαδή οι λογαριασμοί με τους οποίους θα μπορείτε να κάνετε Login θα είναι γνωστοί στην SQL (το λειτουργικό δεν έχει επίγνωση ύπαρξης αυτών των λογαριασμών).

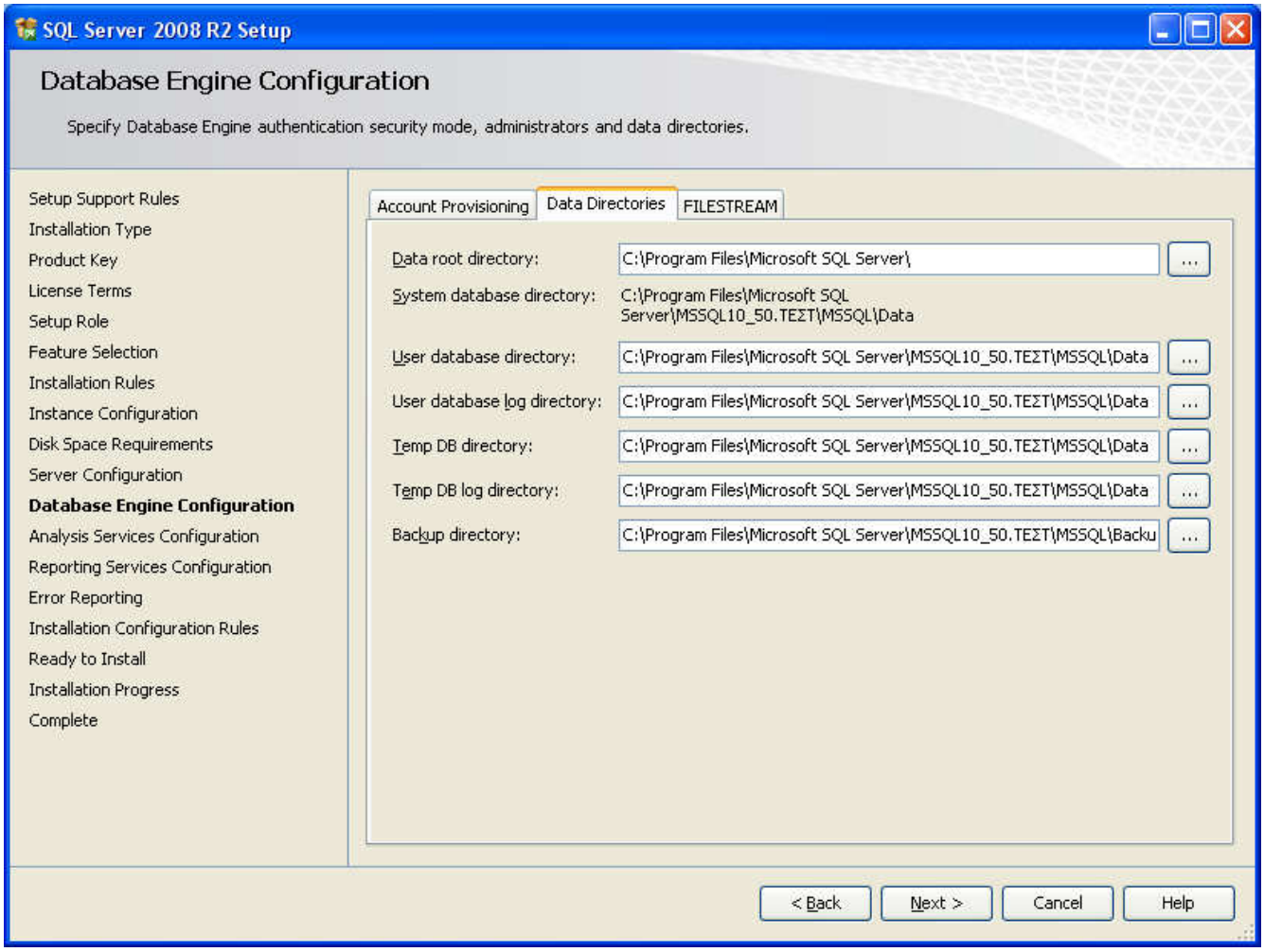
Από Default δημιουργείται ο “sa” λογαριασμός για το Instance, του οποίου το password πρέπει να ορίσετε. Ο λογαριασμός αυτός συνδέεται μόνο με SQL authentication mode (επειδή δεν υπάρχει αυτός ο λογαριασμός στον υπολογιστή/server). Επιπρόσθετα, εισάγετε τουλάχιστον έναν λογαριασμό των Windows ως διαχειριστή (πατήσετε Add Current User για να εισάγεται τον λογαριασμό σας).



Εικόνα 35: *Εγκατάσταση MSSQL Server - DB Engine Conf. - Account Provisioning*

Βήμα 13.

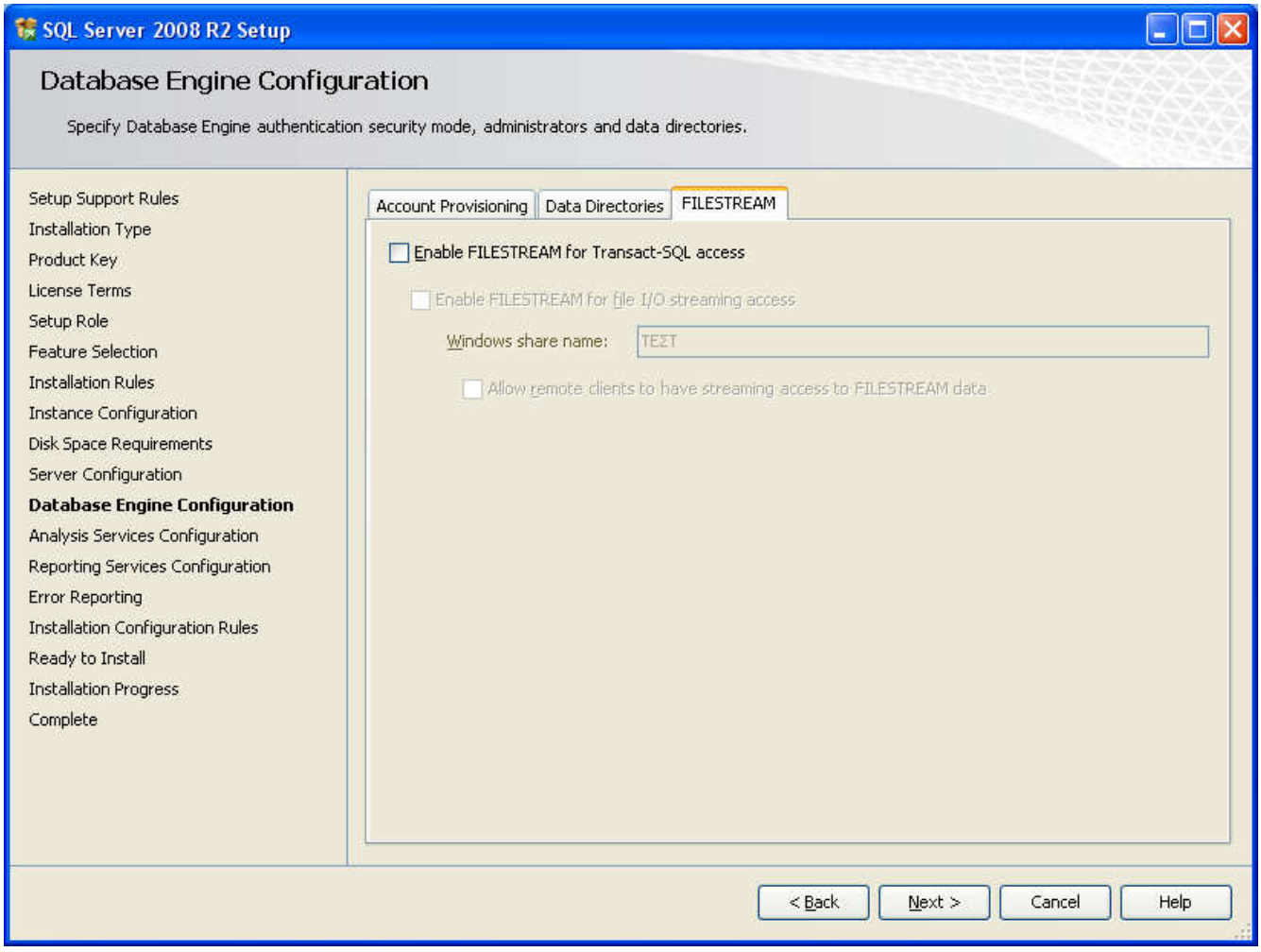
Στην δεύτερη καρτέλα μπορείτε να αλλάξετε τις διευθύνσεις όπου καταχωρούνται τα δεδομένα, τα log files κα.



Εικόνα 36: *Εγκατάσταση MSSQL Server - DB Engine Conf. - Data Directories*

Βήμα 14.

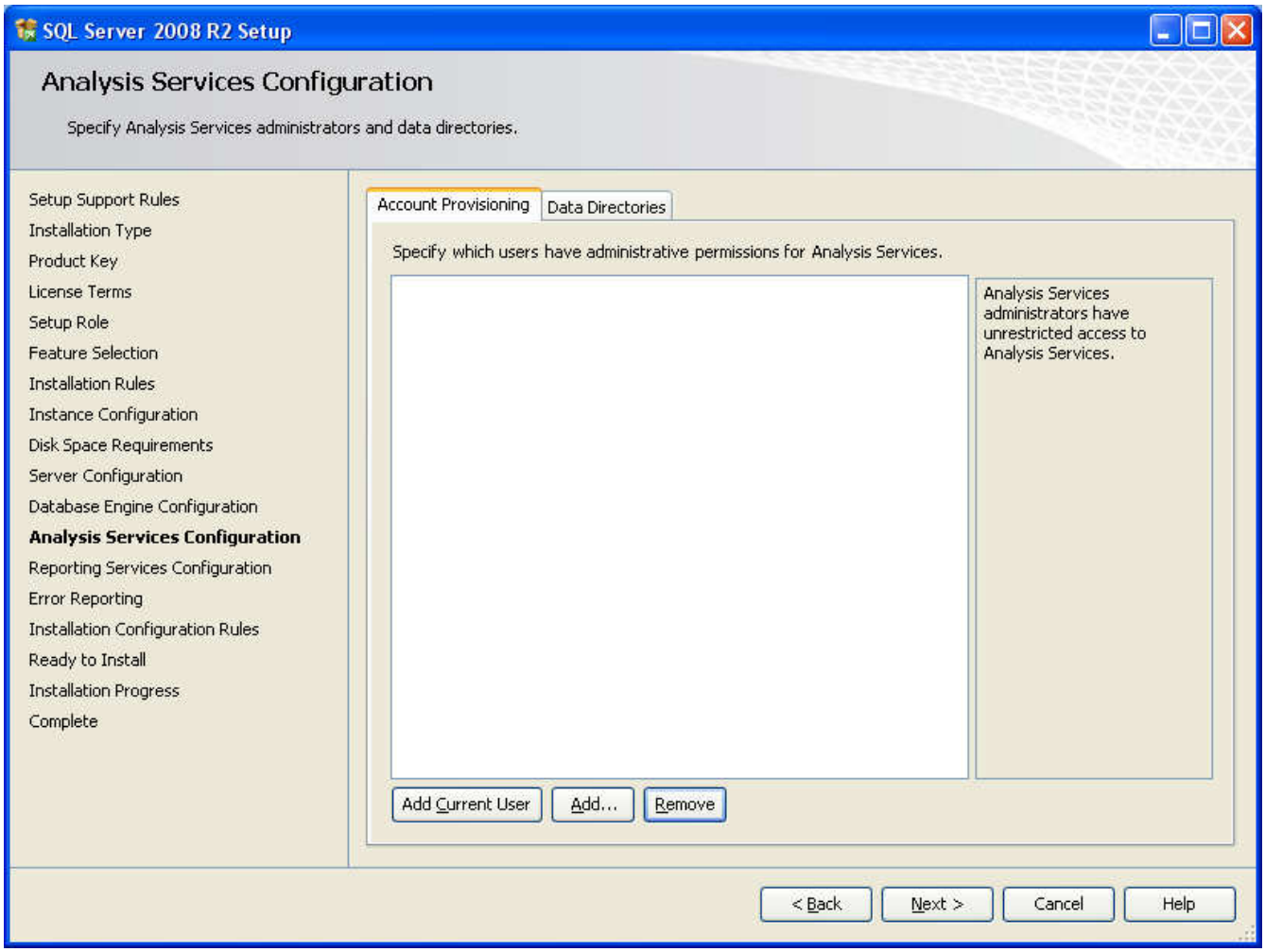
Στην τρίτη καρτέλα δύνεται η δυνατότητα ενεργοποίησης του Filestream (κάτι το οποίο μπορεί να γίνει ανά πάσα στιγμή αργότερα). Το Filestream δίνει την δυνατότητα καταχώρησης μεγάλου όγκου πληροφοριών σε πεδίο πίνακα, όπως για παράδειγμα την εισαγωγή αρχείου εικόνας, ήχου, βίντεο, PDF κα.



Εικόνα 37: *Εγκατάσταση MSSQL Server - DB Engine Conf. - Filestream*

Βήμα 15.

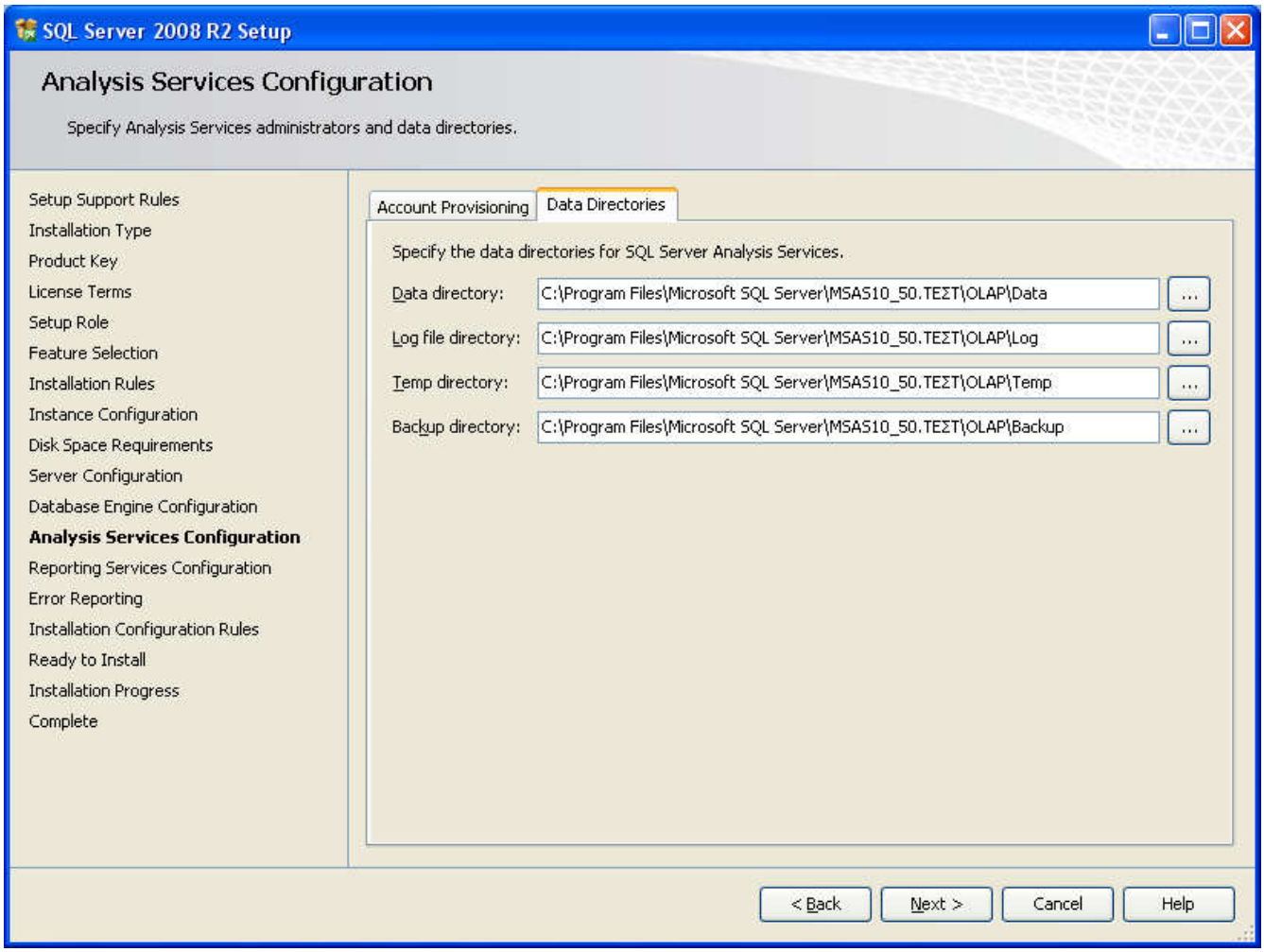
Όμοια, προσθέτουμε τους διαχειριστές για τα Analysis Services.



Εικόνα 38: *Εγκατάσταση MSSQL Server - Analysis Services Conf. - Account Provisioning*

Βήμα 16.

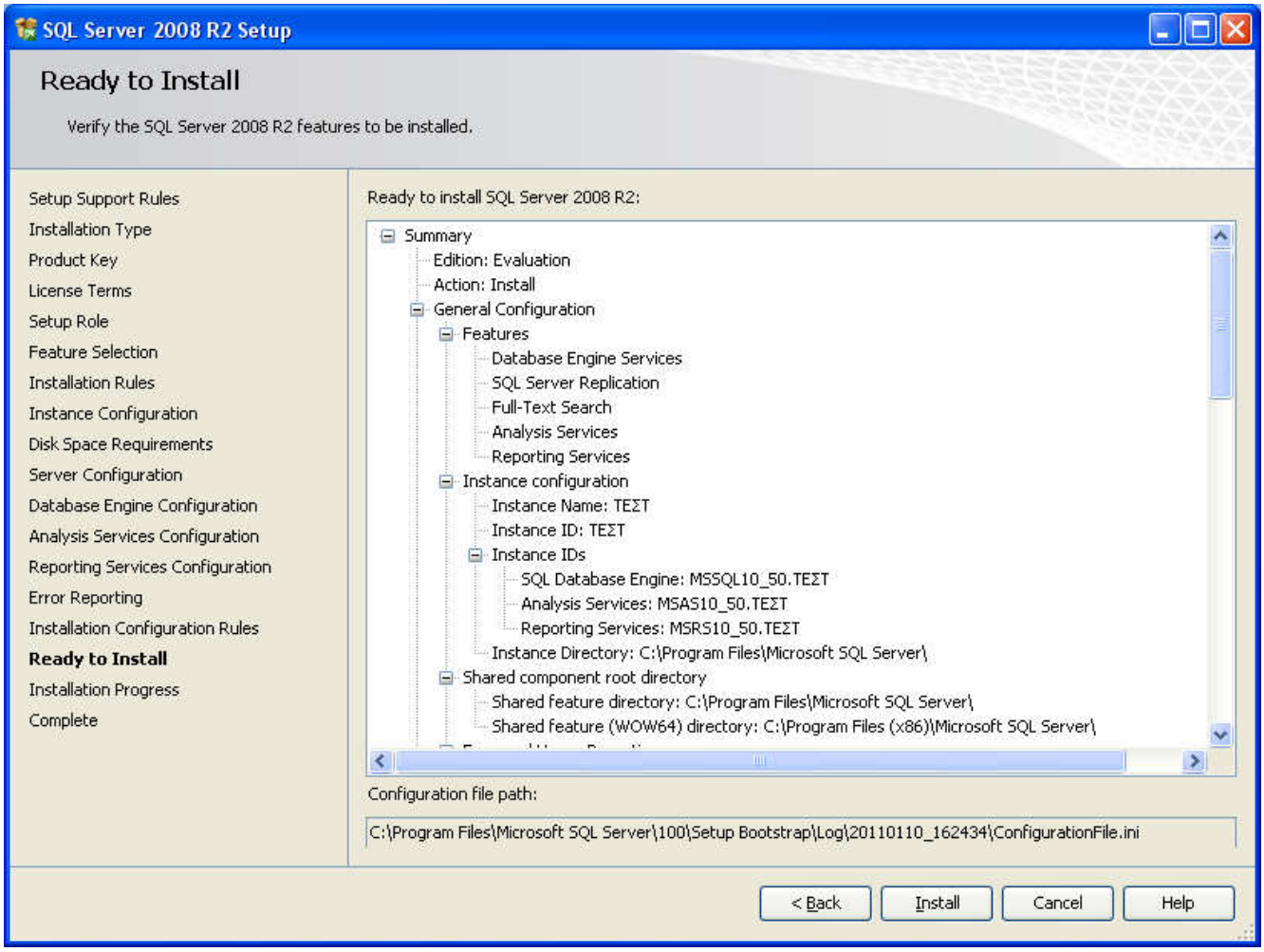
Ρυθμίζουμε τις διευθύνσεις.



Εικόνα 39: *Εγκατάσταση MSSQL Server - Analysis Services Conf. - Data Directories*

Βήμα 17.

Στο τέλος εμφανίζεται ένα συγκεντρωτικό δέντρο με τις επιλογές που έχουμε κάνει και δεν έχουμε παρά να το αποδεχτούμε και να ξεκινήσει η εγκατάσταση.

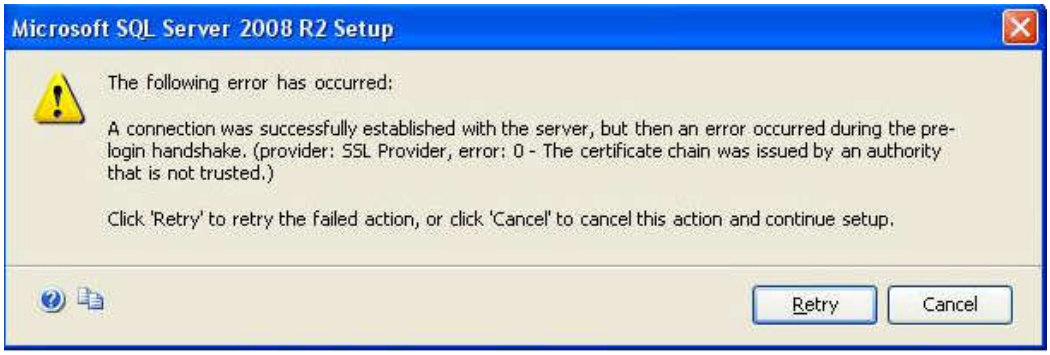


Εικόνα 40: *Εγκατάσταση MSSQL Server – Ready to Install*

Troubleshooting

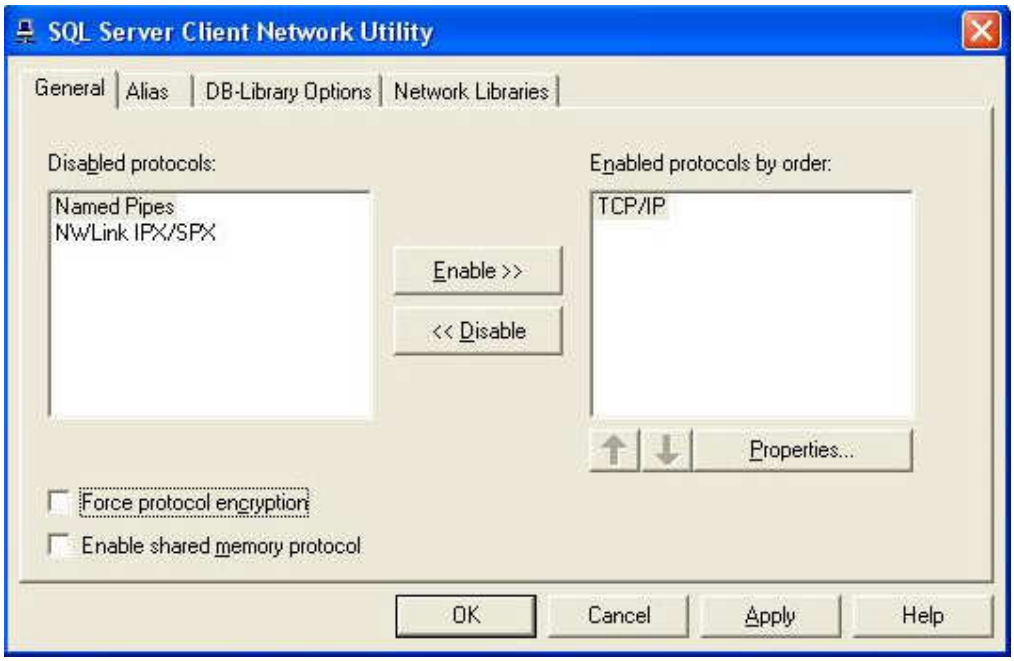
- **Εγκατάσταση**

Στην περίπτωση που προσπαθήσετε να κάνετε την εγκατάσταση πιθανώς να έρθετε αντιμέτωποι με το ακόλουθο error:



Εικόνα 41: *Installation Error Message*

Για να το αντιμετωπίσετε πληκτρολογήστε “cliconfg.exe” σε command line όταν εμφανιστεί το error και θα εμφανιστεί το ακόλουθο παράθυρο.



Εικόνα 42: *SQL Server Client Network Utility*

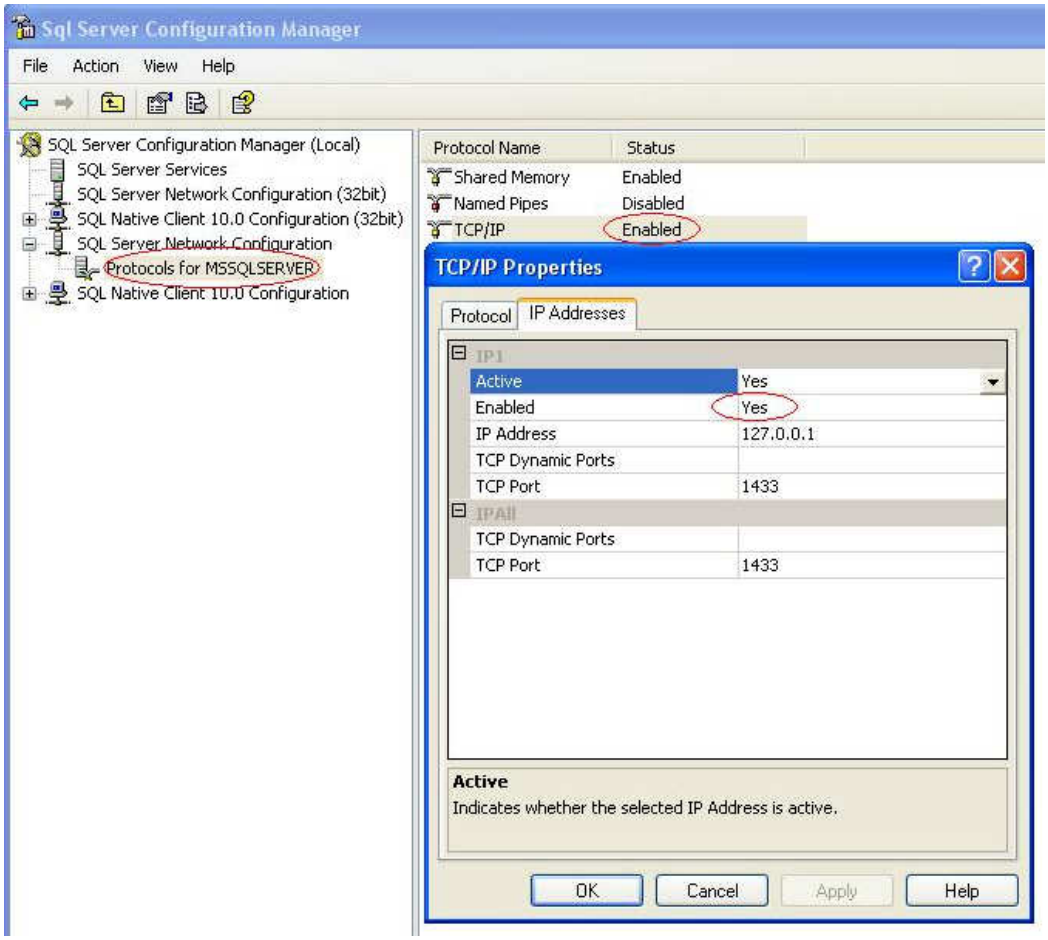
Απεπιλέξτε την επιλογή Force protocol encryption, πατήστε OK, έπειτα Retry στο error message και συνεχίστε την εγκατάσταση κανονικά.

- **Σύνδεση εφαρμογής/προγράμματος με την βάση**

Στην περίπτωση που προσπαθήσετε να συνδεθείτε μέσω δικτύου σε κάποια βάση μέσω προγράμματος, πιθανότατα θα εμφανιστεί το ακόλουθο error:

The TCP/IP connection to the host localhost, port 1433 has failed. Error: "Connection refused: connect. Verify the connection properties, check that an instance of SQL Server is running on the host and accepting TCP/IP connections at the port, and that no firewall is blocking TCP connections to the port."

Σε αυτήν την περίπτωση θα πρέπει να κάνετε τις σημειωμένες ρυθμίσεις στον SQL Server Configuration Manager που δείχνει η *Εικόνα 43: TCP/IP Properties*.



Εικόνα 43: *TCP/IP Properties*

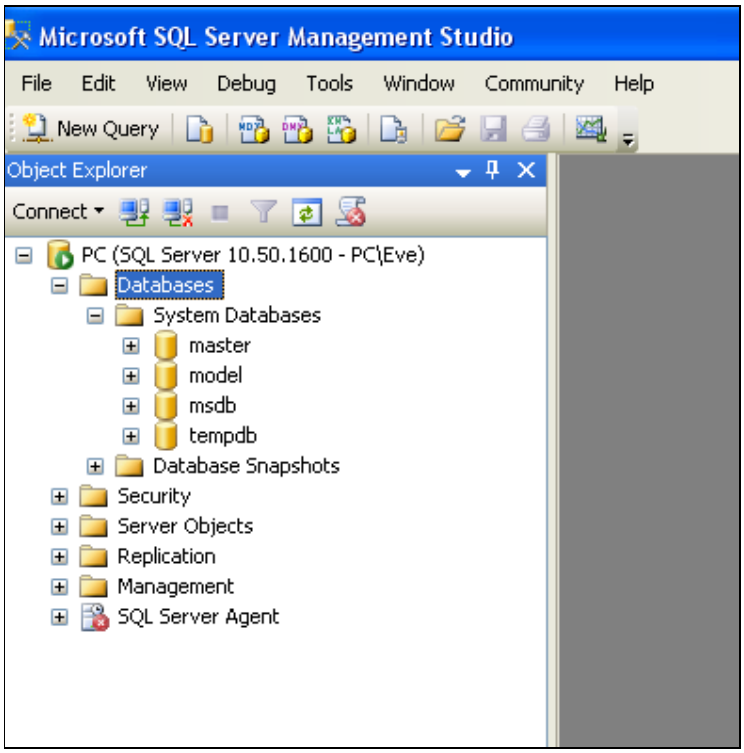
Παράρτημα Δ

Δημιουργία βάσης δεδομένων με χρήση SQL Server Management Studio.

Ο SQL Server Management Studio είναι ο client με τον οποίο διαχειριζόμαστε τις βάσεις. Για να ανοίξουμε το πρόγραμμα επιλέγουμε Start → All Programs → Microsoft SQL Server R2 → SQL Server Management Studio. Στη συνέχει πρέπει να επιλέξουμε Database Engine, στο Server name θα πρέπει να αναγράφεται το όνομα του server/υπολογιστή όπου τρέχει το service και να επιλέξουμε authentication type με βάση το οποίο θέλουμε να συνδεθούμε. Αν για παράδειγμα έχετε μπει στα Windows με λογαριασμό τον οποίο έχετε δηλώσει κατά την εγκατάσταση, μπορείτε απλά να επιλέξετε Windows authentication και να πατήσετε Connect. Διαφορετικά, επιλέξτε SQL authentication και γράψτε στο Username 'sa' και στο Password τον κωδικό που δώσατε κατά την εγκατάσταση.

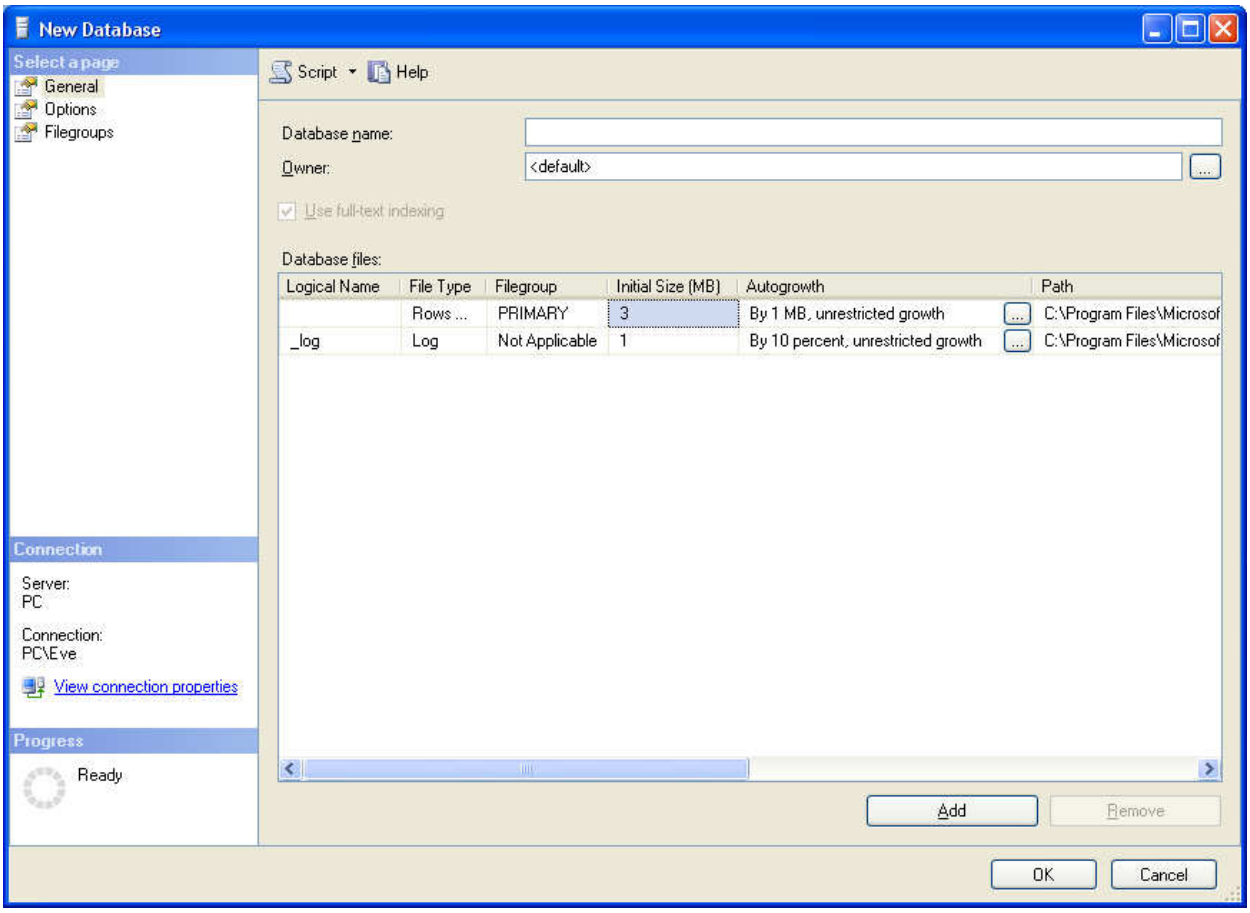
Εικόνα 44: *Connection Window*

Αφού συνδεθούμε κάνουμε δεξί click στο Databases από το αριστερό menu και επιλέγουμε New Database.



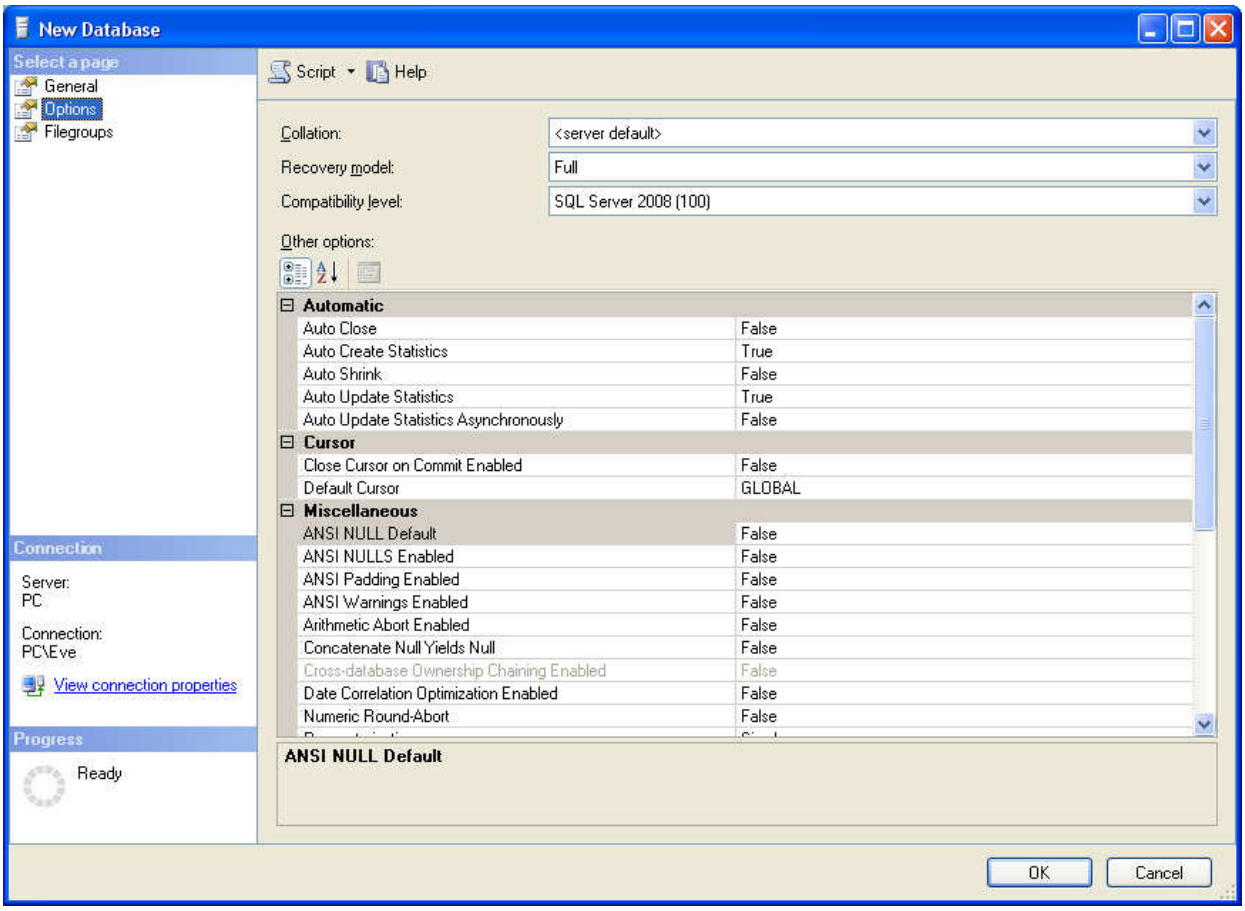
Εικόνα 45: *SQL Management Studio - Object Explorer*

Στο νέο παράθυρο (βλ. *Εικόνα 46: New Database - General*) συμπληρώνουμε το όνομα της βάσης. Αυτόματα συμπληρώνονται και τα Logical Names των αρχείων της βάσης. Το πρώτο αρχείο είναι η βάση αυτή καθ' αυτή. Στο πεδίο Initial Size συμπληρώνουμε το αρχικό μέγεθος που θέλουμε να έχει η βάση. Αν ξέρουμε το μέγεθος που θα έχει η βάση καλόν είναι να το ορίσουμε. Έτσι όλα τα δεδομένα της βάσης θα είναι σε ένα σημείο στον δίσκο και θα αποφύγουμε τον κατακερματισμό. Αν η βάση μας χρειαστεί να υπερβεί το μέγεθος που έχουμε ορίσει τότε αυτόματα θα το αυξήσει ανάλογα με το πόσο έχουμε ορίσει στο πεδίο Autogrowth. Ο κίνδυνος αυτής της αυτόματης προσαύξησης είναι η βάση μας τελικά να καταλάβει όλο το χώρο του δίσκου. Αυτό είναι πολύ συχνό φαινόμενο για τον δεύτερο στη σειρά τύπου αρχείου της βάσης, που είναι το Log file. Όπως ξέρουμε τα log files καταγράφουν κάθε αλλαγή της βάσης. Έτσι το μέγεθος αυτού του αρχείου μεγαλώνει με πολύ μεγάλη ταχύτητα και είναι καλύτερα να ορίσουμε το μέγιστο μέγεθος που επιτρέπετε να έχει.



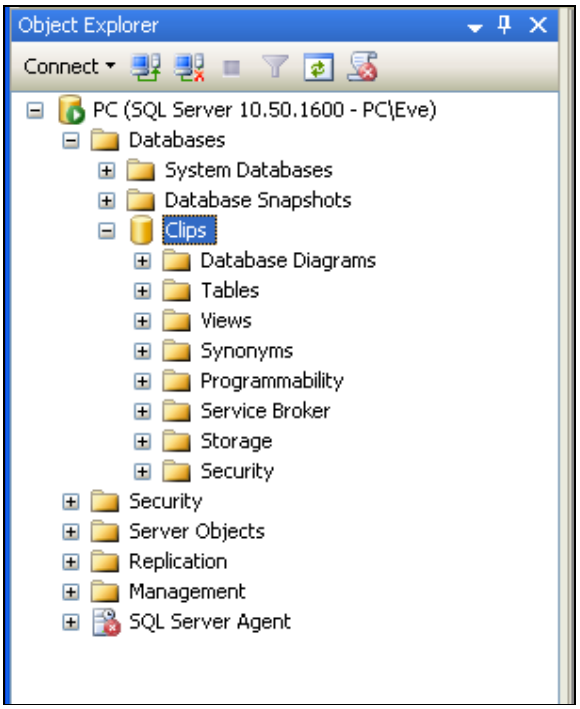
Εικόνα 46: *New Database - General*

Στην δεύτερη καρτέλα (βλ. *Εικόνα 47: New Database - Options*) δίνεται η δυνατότητα να επιλέξουμε το Recovery model που θα έχει η βάση μας. Αν επιλέξουμε Full έχουμε την δυνατότητα να ανακτήσουμε την βάση μας σε οποιαδήποτε χρονική στιγμή κάνοντας restore το backup το οποίο έχουμε πάρει. Αν επιλέξουμε Simple mode, η βάση δεν θα κρατάει log files και κάνοντας restore το Backup θα μπορούμε να κάνουμε ανάκτηση μόνο στην μέρα και ώρα που έχουμε πάρει το backup. Η τρίτη επιλογή είναι η Bulk Logged. Με αυτήν την επιλογή η βάση ναι μεν κρατάει log file αλλά όχι για όλες τις ενδιάμεσες καταστάσεις που περνάει η βάση προκειμένου να ολοκληρώσει οποιαδήποτε bulk εργασία. Το Full recovery model είναι το πιο ασφαλές. Για να μην αντιμετωπίσετε όμως πρόβλημα με το μεγάλο μέγεθος των Log files θα πρέπει να γίνεται backup η βάση (δηλαδή η βάση και τα log files). Μέσω του backup τα log files αδειάζουν.



Εικόνα 47: *New Database - Options*

Πατάμε Ok, και η βάση μας εμφανίζεται στον Object Explorer στα αριστερά. Στην προκειμένη περίπτωση την βάση την ονομάσαμε Clips.



Εικόνα 48: *Clips Database*

Δημιουργία πινάκων με SQL εντολές.

Δημιουργία πίνακα PressMedia

```
USE [Clips]
GO

/****** Object:  Table [dbo].[PressMedia]
Author: Evelina Rimpapova
Script Date: 10/04/2011 00:16:12 ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[PressMedia](
    [id] [int] NOT NULL,
    [name] [nvarchar](250) NOT NULL,
    [type] [nvarchar](250) NOT NULL,
 CONSTRAINT [PK_PressMedia_1] PRIMARY KEY CLUSTERED
(
    [id] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF,
IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]

GO
```

Κώδικας 30: *Δημιουργία πίνακα PressMedia*

Δημιουργία πίνακα Keyword

```
USE [Clips]
GO

/****** Object:  Table [dbo].[PressMedia]
Author: Evelina Rimpapova
Script Date: 10/04/2011 00:16:12 ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[Keyword](
    [id] [int] NOT NULL,
    [name] [nvarchar](250) NOT NULL,
 CONSTRAINT [PK_Keyword_id] PRIMARY KEY CLUSTERED
(
    [id] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF,
IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY],
 CONSTRAINT [unique_KWname] UNIQUE NONCLUSTERED
(
    [name] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]

GO
```

Κώδικας 31: *Δημιουργία πίνακα Keyword*

Δημιουργία πίνακα Company

```
USE [Clips]
GO

/****** Object:  Table [dbo].[Issue]
Author: Evelina Rimpapova
Script Date: 10/04/2011 00:15:57 ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[Company](
    [id] [int] IDENTITY(1,1) NOT NULL,
    [code] [nvarchar](50) NOT NULL,
    [name] [nvarchar](250) NOT NULL,
    [contract_expiration_date] [date] NULL,
 CONSTRAINT [PK_Company] PRIMARY KEY CLUSTERED
(
    [id] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF,
IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON)
ON [PRIMARY]
) ON [PRIMARY]

GO
```

Κώδικας 32: *Δημιουργία πίνακα Company*

Δημιουργία πίνακα Issue

```
USE [Clips]
GO

/****** Object:  Table [dbo].[Issue]
Author: Evelina Rimpapova
Script Date: 10/04/2011 00:15:57 ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[Issue](
    [id] [int] IDENTITY(1,1) NOT NULL,
    [pressMedia_id] [int] NOT NULL,
    [publishedDate] [date] NOT NULL,
    [copies] [int] NULL,
 CONSTRAINT [PK_Issue] PRIMARY KEY CLUSTERED
(
    [id] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF,
IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON)
ON [PRIMARY]
) ON [PRIMARY]

GO

ALTER TABLE [dbo].[Issue]
WITH CHECK ADD  CONSTRAINT [FK_Issue_pressMedia_id]
FOREIGN KEY([pressMedia_id])
REFERENCES [dbo].[PressMedia] ([id])
ON DELETE CASCADE
GO

ALTER TABLE [dbo].[Issue] CHECK CONSTRAINT [FK_Issue_pressMedia_id]
GO
```

Κώδικας 33: *Δημιουργία πίνακα Issue*

Δημιουργία πίνακα InterestedIn

```
USE [Clips]
GO

/****** Object:  Table [dbo].[Issue]
Author: Evelina Rimpapova
Script Date: 10/04/2011 00:15:57 ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[InterestedIn](
    [keyword_id] [int] NOT NULL,
    [company_id] [int] NOT NULL,
 CONSTRAINT [PK_InterestedIn] PRIMARY KEY CLUSTERED
(
    [keyword_id] ASC,
    [company_id] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF,
IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON)
ON [PRIMARY]
) ON [PRIMARY]

GO

ALTER TABLE [dbo].[InterestedIn]
WITH CHECK ADD  CONSTRAINT [FK_InterestedIn_company_id]
FOREIGN KEY([company_id])
REFERENCES [dbo].[Company] ([id])
ON DELETE CASCADE
GO

ALTER TABLE [dbo].[InterestedIn]
CHECK CONSTRAINT [FK_InterestedIn_company_id]
GO

ALTER TABLE [dbo].[InterestedIn]
WITH CHECK ADD  CONSTRAINT [FK_InterestedIn_keyword_id]
FOREIGN KEY([keyword_id])
REFERENCES [dbo].[Keyword] ([id])
ON DELETE CASCADE
GO

ALTER TABLE [dbo].[InterestedIn] CHECK CONSTRAINT [FK_InterestedIn_keyword_id]
GO
```

Κώδικας 34: *Δημιουργία πίνακα InterestedIn*

Δημιουργία πίνακα Text

```
USE [Clips]
GO

/****** Object:  Table [dbo].[Issue]
Author: Evelina Rimpapova
Script Date: 10/04/2011 00:15:57 ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[Text](
    [id] [int] NOT NULL,
    [issue_id] [int] NOT NULL,
    [ocr] [nvarchar](max) NOT NULL,
    [pageSize] [nvarchar](50) NOT NULL,
    [pagesNum] [int] NOT NULL,
    [area] [numeric](18, 3) NOT NULL,
    [page] [nvarchar](100) NOT NULL,
 CONSTRAINT [PK_Text] PRIMARY KEY CLUSTERED
(
    [id] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF,
IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]

GO

ALTER TABLE [dbo].[Text]  WITH CHECK ADD  CONSTRAINT [FK_Text_issue_id] FOREIGN KEY([issue_id])
REFERENCES [dbo].[Issue] ([id])
ON DELETE CASCADE
GO

ALTER TABLE [dbo].[Text] CHECK CONSTRAINT [FK_Text_issue_id]
GO
```

Κώδικας 35: *Δημιουργία πίνακα Text*

Δημιουργία πίνακα LoginUser

```
USE [Clips]
GO

/****** Object:  Table [dbo].[Issue]
Author: Evelina Rimpapova
Script Date: 10/04/2011 00:15:57 ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[LoginUser](
    [id] [int] IDENTITY(1,1) NOT NULL,
    [company_id] [int] NOT NULL,
    [person_fullName] [nvarchar](250) NOT NULL,
    [username] [nvarchar](250) NOT NULL,
    [password] [nvarchar](250) NOT NULL,
    [town] [nvarchar](250) NULL,
    [email] [nvarchar](250) NOT NULL,
 CONSTRAINT [PK_LoginUser] PRIMARY KEY CLUSTERED
(
    [id] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF,
IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON)
ON [PRIMARY],
 CONSTRAINT [unique_username] UNIQUE NONCLUSTERED
(
    [username] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF,
IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON)
ON [PRIMARY]
) ON [PRIMARY]

GO

ALTER TABLE [dbo].[LoginUser]
WITH CHECK ADD  CONSTRAINT [FK_LoginUser_company_id]
FOREIGN KEY([company_id])
REFERENCES [dbo].[Company] ([id])
ON DELETE CASCADE
GO

ALTER TABLE [dbo].[LoginUser] CHECK CONSTRAINT [FK_LoginUser_company_id]
GO
```

Κώδικας 36: *Δημιουργία πίνακα LoginUser*

Δημιουργία πίνακα Clip

```
USE [Clips]
GO
/****** Object:  Table [dbo].[Issue]
Author: Evelina Rimpapova
Script Date: 10/04/2011 00:15:57 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Clip](
    [filename] [int] NOT NULL,
    [company_id] [int] NOT NULL,
    [text_id] [int] NOT NULL,
    [keyword_id] [int] NOT NULL,
    [clippedDatetime] [datetime] NOT NULL,
 CONSTRAINT [PK_Clip_1] PRIMARY KEY CLUSTERED
(
    [filename] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF,
IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]

GO
ALTER TABLE [dbo].[Clip]
WITH CHECK ADD  CONSTRAINT [FK_Clip_company_id] FOREIGN KEY([company_id])
REFERENCES [dbo].[Company] ([id])
ON DELETE CASCADE
GO
ALTER TABLE [dbo].[Clip] CHECK CONSTRAINT [FK_Clip_company_id]
GO
ALTER TABLE [dbo].[Clip]
WITH CHECK ADD  CONSTRAINT [FK_Clip_keyword_id] FOREIGN KEY([keyword_id])
REFERENCES [dbo].[Keyword] ([id])
ON DELETE CASCADE
GO
ALTER TABLE [dbo].[Clip] CHECK CONSTRAINT [FK_Clip_keyword_id]
GO
ALTER TABLE [dbo].[Clip]
WITH CHECK ADD  CONSTRAINT [FK_Clip_text_id] FOREIGN KEY([text_id])
REFERENCES [dbo].[Text] ([id])
ON DELETE CASCADE
GO
ALTER TABLE [dbo].[Clip] CHECK CONSTRAINT [FK_Clip_text_id]
GO
```

Κώδικας 37: Δημιουργία πίνακα Clip

Δημιουργία πίνακα Tender

```
USE [Clips]
GO

/****** Object:  Table [dbo].[Issue]
Author: Evelina Rimpapova
Script Date: 10/04/2011 00:15:57 ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[Tender](
    [text_id] [int] NOT NULL,
    [price] [int] NULL,
    [holder] [nvarchar](250) NULL,
    [expirationDate] [date] NULL,
 CONSTRAINT [PK_Tender] PRIMARY KEY CLUSTERED
(
    [text_id] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF,
IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON)
ON [PRIMARY]
) ON [PRIMARY]

GO

ALTER TABLE [dbo].[Tender]
WITH CHECK ADD  CONSTRAINT [FK_Tender_text_id] FOREIGN KEY([text_id])
REFERENCES [dbo].[Text] ([id])
ON DELETE CASCADE
GO

ALTER TABLE [dbo].[Tender] CHECK CONSTRAINT [FK_Tender_text_id]
GO
```

Κώδικας 38: *Δημιουργία πίνακα Tender*

Δημιουργία πίνακα Article

```
USE [Clips]
GO

/****** Object:  Table [dbo].[Issue]
Author: Evelina Rimpapova
Script Date: 10/04/2011 00:15:57 ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[Article](
    [text_id] [int] NOT NULL,
    [title] [nvarchar](250) NOT NULL,
    [journalist] [nvarchar](250) NULL,
 CONSTRAINT [PK_Article] PRIMARY KEY CLUSTERED
(
    [text_id] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF,
IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]

GO

ALTER TABLE [dbo].[Article]
WITH CHECK ADD  CONSTRAINT [FK_Article_Text_ID] FOREIGN KEY([text_id])
REFERENCES [dbo].[Text] ([id])
ON DELETE CASCADE
GO

ALTER TABLE [dbo].[Article] CHECK CONSTRAINT [FK_Article_Text_ID]
GO
```

Κώδικας 39: *Δημιουργία πίνακα Article*

Δημιουργία των Triggers.

Ο κώδικας των Triggers παραθέτεται στο κύριο μέρος της πτυχιακής, και συγκεκριμένα απεικονίζεται στις εικόνες:

Δημιουργία των stored procedures.

sp_deactivateCompany

```
-- =============================================
-- Author:        <Evelina Rimpapova>
-- Create date: <Αύγουστος 2011>
-- Description:   <Απενεργοποίηση πελάτη. Αν στο πεδίο
-- contract_expiration_date
-- του πίνακα Company υπάρχει δηλωμένη ημερομηνία,
-- τότε θεωρούμε πως ο πελάτης έχει σταματήσει
-- την συνδρομή του.
-- Η ημερομηνία δηλώνει το πότε σταμάτησε.>
-- =============================================
CREATE PROCEDURE [dbo].[sp_deactivateCompany]
      @code nvarchar(50),

      @errMsg nvarchar(500) OUT,
      @myErrCode int OUT
AS
BEGIN
      SET NOCOUNT ON;
      BEGIN TRY
      update Company set contract_expiration_date =
CONVERT(date,GETDATE()) where code = @code
      Select @errMsg = 'No error',@myErrCode = 0
      END TRY

      BEGIN CATCH
      SELECT @errMsg = ERROR_MESSAGE(), @myErrCode = 1
      END CATCH
END
GO
```

Κώδικας 40: *sp_deactivateCompany*

sp_activateCompany

```
-- =============================================
-- Author:        <Evelina Rimpapova>
-- Create date: <Αύγουστος 2011>
-- Description:   < Μέσα από αυτό το stored procedure μπορούμε να
-- ενεργοποιήσουμε ξανά έναν πελάτη. Αν στο πεδίο
-- contract_expiration_date του πίνακα Company υπάρχει δηλωμένη
-- ημερομηνία,τότε ο πελάτης έχει σταματήσει την συνδρομή του.
-- Η ημερομηνία δηλώνει το πότε σταμάτησε.
-- =============================================
CREATE PROCEDURE [dbo].[sp_activateCompany]
      @code nvarchar(50),

      @errMsg nvarchar(500) OUT,
      @myErrCode int OUT
AS
BEGIN
      SET NOCOUNT ON;
      BEGIN TRY
      update Company set contract_expiration_date = null where code =
@code
      Select @errMsg = 'No error',@myErrCode = 0
      END TRY

      BEGIN CATCH
            SELECT @errMsg = ERROR_MESSAGE(), @myErrCode = 1
      END CATCH
END
      GO
```

Κώδικας 41: *sp_activateCompany*

sp_insertKeyword

```
-- =============================================
-- Author:        <Evelina Rimpapova>
-- Create date: <Αύγουστος 2011>
-- Description:   < Μέσα από αυτό το stored procedure μπορούμε να
-- εισάγουμε πελάτη και τις λέξεις κλειδιά για τις
-- οποίες ενδιαφέρεται. Αν ο πελάτης ή η λέξη κλειδί υπάρχουν,
-- δεν εισάγονται ξανά.
-- =============================================
   USE [Clips]
   GO
   CREATE PROCEDURE [dbo].[sp_insertKeyword]

       @company_code nvarchar(50),
       @company_name varchar(250),

       @keyword_id int,
       @keyword_name nvarchar(250),

       @loginUser_personFullName nvarchar(250),
       @loginUser_email nvarchar(250),
       @loginUser_usernameAndPass nvarchar(250),
       @loginUser_town nvarchar(250),

       @errMsg nvarchar(500) OUT,
       @myErrCode int OUT

   AS
       SET NOCOUNT ON;
       declare @company_id int

       SET NOCOUNT ON;

       IF @@TRANCOUNT > 0
           SAVE TRAN myTran
       ELSE
           BEGIN TRAN
```

Κώδικας 42: *sp_insertKeyword (τμήμα 1)*

```
BEGIN TRY

--Company Info
IF NOT EXISTS (SELECT id FROM Company where code = @company_code)
BEGIN
    Insert into Company(code, name) values (@company_code,@company_name)
    Select @company_id = id from Company where code = @company_code
    Insert into LoginUser(company_id,username,password,person_fullName,town,email)
    values (@company_id,@loginUser_usernameAndPass,@loginUser_usernameAndPass,
    @loginUser_personFullName,@loginUser_town, @loginUser_email)
END
ELSE
BEGIN
    IF NOT EXISTS ( SELECT ID FROM Company
    WHERE code = @company_code AND name = @company_name)
    BEGIN
    UPDATE Company set name = @company_name where code = @company_code
    END
END
Select @company_id = id from Company where code = @company_code

--Keyword info
IF NOT EXISTS (SELECT ID from Keyword where id = @keyword_id)
IF NOT EXISTS (SELECT ID from Keyword where name = @keyword_name)
BEGIN
    Insert into Keyword(id,name) values(@keyword_id,@keyword_name)
END
ELSE
BEGIN
    IF NOT EXISTS
    ( SELECT ID from Keyword where id = @keyword_id AND name = @keyword_name)
    BEGIN
    UPDATE Keyword set name = @keyword_name where id = @keyword_id
    END
END
```

Κώδικας 43: *sp_insertKeyword (τμήμα 2)*

```
--InterestedIn info
IF NOT EXISTS
(SELECT * from InterestedIn
where keyword_id = @keyword_id and company_id = @company_id)
BEGIN
    insert into InterestedIn(keyword_id,company_id)
    values (@keyword_id,@company_id)
END

return (Select id from Company where code = @company_code)

Select @errMsg = 'No error',@myErrCode = 0
END TRY

BEGIN CATCH
    IF @@TRANCOUNT > 1 ROLLBACK TRANSACTION myTran
    ELSE ROLLBACK TRANSACTION
    SELECT @errMsg = ERROR_MESSAGE(), @myErrCode = 1
END CATCH

IF @@TRANCOUNT > 1 COMMIT TRANSACTION myTran
ELSE COMMIT TRANSACTION;
```

Κώδικας 44: *sp_insertKeywrod (τμήμα 3)*

sp_insertArticle

```
-- =============================================
-- Author:      <Rimpapova Evelina>
-- Create date: <2011>
-- Description: <Procedure για την εισαγωγή άρθρων στη βάση.>
-- =============================================

CREATE PROCEDURE [dbo].[sp_insertArticle]

    @pressMedia_id int,
    @pressMedia_name nvarchar(250),
    @pressMedia_type nvarchar(250),

    @issue_publishedDate date,

    @issue_copies int,

    @text_id int,
    @text_ocr nvarchar(max),
    @text_pageSize nvarchar(2),
    @text_pagesNum int,
    @text_area numeric(18,3),
    @text_page nvarchar(100),

    @article_title nvarchar(250),
    @article_journalist nvarchar(250),

    @clip_filename int,
    @clip_clippedDatetime datetime,


    @company_code nvarchar(50),
    @company_name nvarchar(250),

    @keyword_id int,
    @keyword_name nvarchar(250),

    @loginUser_personFullName nvarchar(250),
    @loginUser_email nvarchar(250),
    @loginUsernameAndPass nvarchar(250),
    @loginUser_town nvarchar(250),

    @errMsg nvarchar(500) OUT,
    @myErrCode int OUT
```

Κώδικας 45: *sp_insertArticle (τμήμα 1)*

```
AS
    declare @issue_id int
    declare @company_id int
    SET NOCOUNT ON;

    BEGIN TRANSACTION myTran;

    BEGIN TRY

        --Press Media info
        IF NOT EXISTS
        (SELECT ID FROM PressMedia WHERE ID = @pressMedia_id)
        BEGIN
            Insert into PressMedia(id,name,type)
            values(@pressMedia_id,@pressMedia_name,@pressMedia_type)
        END
        ELSE
        BEGIN
            IF NOT EXISTS
            ( SELECT ID FROM PressMedia
            WHERE ID = @pressMedia_id AND name = @pressMedia_name
            AND type = @pressMedia_type)
            BEGIN
            UPDATE PressMedia set name=@pressMedia_name,
            type=@pressMedia_type where id=@pressMedia_id
            END
        END

        --Issue info
        IF NOT EXISTS
        (SELECT id FROM Issue
        WHERE pressMedia_id = @pressMedia_id
        and publishedDate = @issue_publishedDate)
        BEGIN
            Insert into Issue(pressMedia_id,publishedDate,copies)
            values(@pressMedia_id,@issue_publishedDate,@issue_copies)
        END
        select @issue_id=id
        from Issue where pressMedia_id = @pressMedia_id
        and publishedDate = @issue_publishedDate
```

Κώδικας 46: *sp_insertArticle (τμήμα 2)*

```
    --Article info
    IF NOT EXISTS (SELECT id FROM Text WHERE id = @text_id)
    BEGIN
        Insert into Text(id,issue_id,ocr,pageSize,pagesNum,area,page)
        values(@text_id,@issue_id,@text_ocr,@text_pageSize,
        @text_pagesNum,@text_area,@text_page)
        insert into Article(text_id,title,journalist)
        values(@text_id, @article_title, @article_journalist)
    END

    DECLARE @errMsg2 nvarchar(500)
    DECLARE @myErrCode2 INT
    EXEC @company_id = sp_insertKeyword @company_code, @company_name ,
        @keyword_id , @keyword_name , @loginUser_personFullName ,
        @loginUser_email,@loginUsernameAndPass, @loginUser_town,
        @errMsg2 OUTPUT, @myErrCode2 OUTPUT
    if(@myErrCode2<>0)
    BEGIN
        SELECT @errMsg = @errMsg2, @myErrCode = 1
        return
    END

    --Clip Info
    IF NOT EXISTS
    (SELECT filename FROM Clip WHERE filename = @clip_filename )
    BEGIN
        Insert into
        Clip(filename,company_id,text_id,keyword_id,clippedDatetime)
        values (@clip_filename,@company_id, @text_id,
        @keyword_id, @clip_clippedDatetime)
    END
    Select @errMsg = 'No error',@myErrCode = 0
END TRY

BEGIN CATCH
    ROLLBACK TRANSACTION myTran
    SELECT @errMsg = ERROR_MESSAGE(), @myErrCode = 1
END CATCH

COMMIT TRANSACTION myTran;
```

Κώδικας 47: *sp_insertArticle (τμήμα 3)*

sp_insertTender

```
-- ================================================
-- Author:      <Rimpapova Evelina>
-- Create date: <2011>
-- Description: <Procedure για την εισαγωγή δημοπρασιών στη βάση.>
-- ================================================
CREATE PROCEDURE [dbo].[sp_insertTender]

    @pressMedia_id int,
    @pressMedia_name nvarchar(250),
    @pressMedia_type nvarchar(250),

    @issue_publishedDate date,

    @issue_copies int,

    @text_id int,
    @text_ocr nvarchar(max),
    @text_pageSize nvarchar(2),
    @text_pagesNum int,
    @text_area numeric(18,3),
    @text_page nvarchar(100),

    @tender_price int,
    @tender_holder nvarchar(250),
    @tender_expirationDate date,

    @clip_filename int,
    @clip_clippedDatetime datetime,


    @company_code nvarchar(50),
    @company_name nvarchar(250),

    @keyword_id int,
    @keyword_name nvarchar(250),

    @loginUser_personFullName nvarchar(250),
    @loginUser_email nvarchar(250),
    @loginUsernameAndPass nvarchar(250),
    @loginUser_town nvarchar(250),

    @errMsg nvarchar(500) OUT,
    @myErrCode int OUT
```

Κώδικας 48: *sp_insertTender (τμήμα 1)*

```
AS
    declare @issue_id int
    declare @company_id int

    SET NOCOUNT ON;
    BEGIN TRANSACTION myTran;
    BEGIN TRY

        --Press Media info
        IF NOT EXISTS
        (SELECT ID FROM PressMedia WHERE ID = @pressMedia_id)
        BEGIN
            Insert into
            PressMedia(id,name,type)
            values(@pressMedia_id,@pressMedia_name,@pressMedia_type)
        END
        ELSE
        BEGIN
            IF NOT EXISTS
            ( SELECT ID FROM PressMedia
            WHERE ID = @pressMedia_id AND name = @pressMedia_name
            AND type = @pressMedia_type)
            BEGIN
            UPDATE PressMedia set name=@pressMedia_name,
            type=@pressMedia_type where id=@pressMedia_id
            END
        END

        --Issue info
        IF NOT EXISTS
        (SELECT id FROM Issue
        WHERE pressMedia_id = @pressMedia_id
        and publishedDate = @issue_publishedDate)
        BEGIN
            Insert into Issue(pressMedia_id,publishedDate,copies)
            values(@pressMedia_id,@issue_publishedDate,@issue_copies)
        END
        select @issue_id=id from Issue
        where pressMedia_id = @pressMedia_id
        and publishedDate = @issue_publishedDate
```

Κώδικας 49: *sp_insertTender (τμήμα 2)*

```
--Tender info
IF NOT EXISTS (SELECT id FROM Text WHERE id = @text_id)
    BEGIN
        Insert into
        Text(id,issue_id,ocr,pageSize,pagesNum,area,page)
        values(@text_id,@issue_id,@text_ocr,@text_pageSize,
        @text_pagesNum,@text_area,@text_page)
        insert into
        Tender(text_id,price,holder,expirationDate)
        values(@text_id, @tender_price,
        @tender_holder, @tender_expirationDate)
END

DECLARE @errMsg2 nvarchar(500)
DECLARE @myErrCode2 INT
EXEC @company_id = sp_insertKeyword @company_code, @company_name ,
    @keyword_id , @keyword_name , @loginUser_personFullName ,
    @loginUser_email, @loginUsernameAndPass, @loginUser_town,
    @errMsg2 OUTPUT, @myErrCode2 OUTPUT
if(@myErrCode2<>0)
BEGIN
    SELECT @errMsg = @errMsg2, @myErrCode = 1
    return
END

--Clip Info
IF NOT EXISTS
(SELECT filename FROM Clip WHERE filename = @clip_filename )
BEGIN
    Insert
    into Clip(filename,company_id,text_id,keyword_id,clippedDatetime)
    values (@clip_filename,@company_id, @text_id,
    @keyword_id, @clip_clippedDatetime)
END

Select @errMsg = 'No error',@myErrCode = 0
END TRY

BEGIN CATCH
    ROLLBACK TRANSACTION myTran
    SELECT @errMsg = ERROR_MESSAGE(), @myErrCode = 1
END CATCH

COMMIT TRANSACTION myTran;
```

Κώδικας 50: *sp_insertTender (τμήμα 3)*

Παράρτημα Ε

Ακολουθεί ο κώδικας της Java Εφαρμογής που περιγράφεται στην παρούσα πτυχιακή. Όλες οι κλάσεις, πλην των LastModifiedFileComparator.java, PdfExtensionFilter.java, StdOutLevel.java και LoggingOutputStream.java (οι οποίες όμως έχουν τροποποιηθεί ανάλογα), είναι αποτέλεσμα προσωπικής εργασίας.

ConfigurationContentHandler.java

```
package pdf2djvu;

import org.xml.sax.*;
import javax.swing.JOptionPane;


public class ConfigurationContentHandler implements ContentHandler {

  private int interval;
  private String general_input;
  private String redirectionPath;
  private int ifFilesMoreThanLimit;
  private String pdf_output;
  private String output_onError;
  private String djvu_output;
  private String xml_output;
  private String logFiles[][] = new String[3][4];
    private String logAttr_maxSizeInMB;
    private String logAttr_numberOfLogFiles;
    private String logAttr_append;

  private EmailContainer emailCont = null;
  private EmailFields tmpEmail = new EmailFields();
  private int email_index=0;
  private int logFiles_number=0;

  private String SQLip;
  private String SQLdatabaseName;
  private String SQLusername;
  private String SQLpassword;

  private String OracleIp;
  private String OracleDatabaseName;
  private String OracleUsername;
  private String OraclePassword;
  private String OracleInernalLogon;

  private String convertCommand1;
  private String convertProfile;

  private String convertCommand2;
  private String[] options;
```

```
  private int numberOfThreads = 2;

  private String oracle_mssql_synch_enabled;
  private String oracle_mssql_synch_start_time;
  private String oracle_mssql_synch_interval;

  private String tmp="";
  private int i=0;

  public void characters(char[] text, int start, int length)
   throws SAXException {
      tmp=String.valueOf(text,start,length);
  }


  public void startElement(String namespaceURI, String localName,
   String qualifiedName, Attributes atts) {
        if(qualifiedName.equals("general_input")){
            interval =
Integer.parseInt(atts.getValue("interval")+"000");//μετατροπή σε
second
        }
        if(qualifiedName.equals("redirectionPath")){
            ifFilesMoreThanLimit =
Integer.parseInt(atts.getValue("ifFilesMoreThan"));
        }
        if(qualifiedName.equals("email_config")){
            i=0;
             emailCont = new EmailContainer();

        }
        if(qualifiedName.equals("log")){
            i=0;
            logFiles_number++;
            logAttr_maxSizeInMB = atts.getValue("maxSizeInMB");
            logAttr_numberOfLogFiles =
atts.getValue("numberOfLogFiles");
            logAttr_append = atts.getValue("append");
        }
  }

   public void endElement(String namespaceURI, String localName,
   String qualifiedName) {
      if(qualifiedName.equals("general_input")){
          general_input = tmp;
      }
      if(qualifiedName.equals("redirectionPath")){
          redirectionPath = tmp;
      }
      if(qualifiedName.equals("pdf_output")){
          pdf_output = tmp;
      }
      if(qualifiedName.equals("output_onError")){
          output_onError = tmp;
      }
      if(qualifiedName.equals("djvu_output")){
          djvu_output = tmp;
      }
      if(qualifiedName.equals("xml_output")){
          xml_output = tmp;
```

```
        }
        if(qualifiedName.equals("log")){
            while(logFiles[i][0] != null){
                i++;
            }
            try{
              logFiles[i][0] = tmp;
              logFiles[i][1] = logAttr_maxSizeInMB;
              logFiles[i][2] = logAttr_numberOfLogFiles;
              logFiles[i][3] = logAttr_append;
            }
            catch(IndexOutOfBoundsException e){
              System.out.println("IndexOutOfBoundsException. 'Log'
field.");
               System.out.println(e.getMessage());
            }
        }
        if(qualifiedName.equals("log_files")){
            if(logFiles_number < logFiles.length){
                    JOptionPane.showMessageDialog(null, "A log file
name configuration is missing. Please stop the application and add a
log file name\n\r in Configuration file or error will
occur!","Error!",JOptionPane.ERROR_MESSAGE);
            }
        }
        if(qualifiedName.equals("email")){
            emailCont.setEmail(tmpEmail);
            tmpEmail = new EmailFields();
            email_index++;
        }
        if(qualifiedName.equals("host")){
            tmpEmail.setHost(tmp);
        }
        if(qualifiedName.equals("from")){
            tmpEmail.setFrom(tmp);
        }
        if(qualifiedName.equals("to")){
            String[] to = new String[tmpEmail.getTo().length];
            to = tmpEmail.getTo();
            while(to[i]!=null){
                i++;
            }
            try{
              to[i] = tmp.trim();
              tmpEmail.setTo(to);
            }
            catch(IndexOutOfBoundsException e){
              System.out.println("IndexOutOfBoundsException. 'To'
field.");
               System.out.println(e.getMessage());
            }

        }
        if(qualifiedName.equals("subject")){
            tmpEmail.setSubject(tmp);
        }
        if(qualifiedName.equals("body")){
            tmpEmail.setBody(tmp);
        }
        if(qualifiedName.equals("body_sample_file")){
            tmpEmail.setBody_sample_file(tmp);
```

```
        }
        if(qualifiedName.equals("IP")){
            SQLip = tmp;
        }
        if(qualifiedName.equals("databaseName")){
            SQLdatabaseName = tmp;
        }
        if(qualifiedName.equals("username")){
            SQLusername = tmp;
        }
        if(qualifiedName.equals("password")){
            SQLpassword = tmp;
        }
        if(qualifiedName.equals("command")){
            convertCommand1 = tmp;
        }
        if(qualifiedName.equals("profile")){
            convertProfile = tmp;
        }
        if(qualifiedName.equals("programPath")){
            convertCommand2 = tmp;
        }
        if(qualifiedName.equals("options")){
            tmp = tmp.replaceAll("   ", " ");
            tmp = tmp.replaceAll("  ", " ");
            options = tmp.split(" ");
        }
        if(qualifiedName.equals("number")){
            numberOfThreads = Integer.parseInt(tmp);
        }
        if(qualifiedName.equals("OracleIP")){
            OracleIp = tmp;
        }
        if(qualifiedName.equals("OracledatabaseName")){
            OracleDatabaseName = tmp;
        }
        if(qualifiedName.equals("Oracleusername")){
            OracleUsername = tmp;
        }
        if(qualifiedName.equals("Oraclepassword")){
            OraclePassword = tmp;
        }
        if(qualifiedName.equals("OracleInternalLogon")){
            OracleInernalLogon = tmp;
        }
        if(qualifiedName.equals("enabled")){
            oracle_mssql_synch_enabled = tmp;
        }
        if(qualifiedName.equals("start_time")){
            oracle_mssql_synch_start_time = tmp;
        }
        if(qualifiedName.equals("interval")){
            oracle_mssql_synch_interval = tmp;
        }
    }

    // do-nothing methods
    public void setDocumentLocator(Locator locator) {}
    public void startDocument() {}
    public void endDocument() {}
    public void startPrefixMapping(String prefix, String uri) {}
```

```
public void endPrefixMapping(String prefix) {}
public void ignorableWhitespace(char[] text, int start,
int length) throws SAXException {}
public void processingInstruction(String target, String data){}
public void skippedEntity(String name) {}

public int getInterval(){
    return interval;
}
public String getPdf_output(){
   return pdf_output;
}
public String getGeneral_input(){
   return general_input;
}
public int getMoveFilesLimit() {
   return ifFilesMoreThanLimit;
}
public String getRedirectionPath() {
  return redirectionPath;
}
public String getOutput_onError(){
  return output_onError;
}
public String getDjvu_output(){
  return djvu_output;
}
public String getXml_output(){
  return xml_output;
}
public String[][] getLogFiles(){
  return logFiles;
}
public int getLogFiles_number(){
    return logFiles_number;
}
public EmailContainer getEmailCont(){
    return emailCont;
}

  public String getconvertCommand1() {
      return convertCommand1;
  }

  public String getConvertProfile() {
      return convertProfile;
  }

  public String getConvertCommand2() {
      return convertCommand2;
  }

  public String[] getOptions() {
      return options;
  }

  public String getSQLdatabaseName() {
      return SQLdatabaseName;
  }

  public String getSQLip() {
```

```
        return SQLip;
    }

    public String getSQLpassword() {
        return SQLpassword;
    }

    public String getSQLusername() {
        return SQLusername;
    }

    public String getOracleDatabaseName() {
        return OracleDatabaseName;
    }

    public String getOracleInernalLogon() {
        return OracleInernalLogon;
    }

    public String getOracleIp() {
        return OracleIp;
    }

    public String getOraclePassword() {
        return OraclePassword;
    }

    public String getOracleUsername() {
        return OracleUsername;
    }

    public int getNumberOfThreads() {
        return numberOfThreads;
    }

    public String getOracle_mssql_synch_enabled() {
        return oracle_mssql_synch_enabled;
    }

    public String getOracle_mssql_synch_interval() {
        return oracle_mssql_synch_interval;
    }

    public String getOracle_mssql_synch_start_time() {
        return oracle_mssql_synch_start_time;
    }
}
```

ReadXMLConf.java

```
package pdf2djvu;

import java.io.IOException;
import org.xml.sax.InputSource;
import org.xml.sax.XMLReader;
import org.xml.sax.helpers.XMLReaderFactory;

public class ReadXMLConf {
    private String source = "file:///";
    private ConfigurationContentHandler contentHandl = null;
    public ReadXMLConf(String src){
        source += src;
        try{
           XMLReader parser = XMLReaderFactory.createXMLReader();
           contentHandl = new ConfigurationContentHandler();

           parser.setContentHandler(contentHandl);
           parser.parse(new InputSource(source));

        }
        catch(org.xml.sax.SAXException e){
            System.out.println("The "+source+" is not well formed.");
        }
        catch(IOException e){
            System.out.println(e.getMessage());
        }

    }
    public String toString(){

        return "Configuration Source: "+ source + "\n\r"
+contentHandl.toString();
    }

    public ConfigurationContentHandler getMyContentHandler(){
        return contentHandl;
    }

}
```

DistributionXmlObj.java

```
package pdf2djvu;

public class DistributionXmlObj {
    private String author;
    private String id;
    private String artId;
    private String headline;
    private String pageName;
    private String numberOfClippings;
    private String articleSize;
    private String articleType;
    private String foreas;
    private String pageSize;
    private String publicationDate;
    private String timeCreated;
    private String searchWord;
    private String searchWordId;
    private String publicationId;
    private String publicationName;
    private String publicationCirc;
    private String paper_type;
    private String customerId;
    private String customerName;
    private String cust_resp;
    private String cust_city;
    private String cust_email = "invalidemail@domain.gr";
    private String articleText;

    private String tenderAmount ;
    private String tenderExpireDate ;



    public DistributionXmlObj() {}

    public void setAuthor(String s){
        author = s;
    }

    public String getAuthor(){
        return author;
    }


    public void setArticleText(String articleText) {
        this.articleText = articleText;
    }

    public void setForeas(String foreas) {
        this.foreas = foreas;
    }


    public String getArtId() {
        return artId;
    }
```

```
public String getArticleSize() {
    return articleSize;
}

public String getArticleText() {
    return articleText;
}

public String getArticleType() {
    return articleType;
}

public String getCust_city() {
    return cust_city;
}

public String getCust_email() {
    return cust_email;
}

public String getCust_resp() {
    return cust_resp;
}

public String getCustomerId() {
    return customerId;
}

public String getCustomerName() {
    return customerName;
}

public String getForeas() {
    return foreas;
}

public String getHeadline() {
    return headline;
}

public String getId() {
    return id;
}

public String getPageName() {
    return pageName;
}

public String getPageSize() {
    return pageSize;
}

public String getNumberOfClippings() {
    return numberOfClippings;
}

public void setNumberOfClippings(String numberOfClippings) {
    this.numberOfClippings = numberOfClippings;
}
```

```
public String getPaper_type() {
    return paper_type;
}

public String getPublicationCirc() {
    return publicationCirc;
}

public String getPublicationDate() {
    return publicationDate;
}

public String getPublicationId() {
    return publicationId;
}

public String getPublicationName() {
    return publicationName;
}

public String getSearchWord() {
    return searchWord;
}

public String getTimeCreated() {
    return timeCreated;
}



public void setId(String id){
    this.id = id;
}
 public void setArtId(String artId){
    this.artId = artId;
}
public void setHeadline(String h){
    headline = h;
}
public void setPageName(String p){
    pageName = p;
}
public void setArticleSize(String a){
    articleSize = a;
}
public void setArticleType(String at){
    articleType = at;
}
public void setPageSize(String p){
    pageSize = p;
}
public void setPublicationDate(String pd){
    publicationDate = pd;
}
public void setTimeCreated(String tc){
    timeCreated = tc;
}
public void setSearchWord(String sw){
    searchWord = sw;
```

```
}
public void setSearchWordId(String searchWordId) {
    this.searchWordId = searchWordId;
}
public void setPublicationId(String id){
    publicationId = id;
}
public void setPublicationName(String n){
    publicationName = n;
}
public void setPublicationCirc(String c){
    publicationCirc = c;
}
public void setPaper_type(String pt){
    paper_type = pt;
}

public void setCustomerId(String id){
    customerId = id;
}
public void setCustomerName(String n){
    customerName = n;
}
public void setCust_resp(String cr){
    cust_resp = cr;
}
public void setCust_city(String cc){
    cust_city = cc;
}
public void setCust_email(String e){
    cust_email = e;
}


public String getTimeCreatedTime(){
    return timeCreated.substring(0,10);
}

public String getTimeCreatedDate(){
    return timeCreated.substring(11);
}

public String getTenderAmount() {
    return tenderAmount;
}


public String getTenderExpireDate() {
    return tenderExpireDate;
}

public String getSearchWordId() {
    return searchWordId;
}


public String[] getCust_emailTable(){
    int pos = cust_email.indexOf(";");
    int num = 1;
    while (pos !=-1){
        num++;
```

```
            pos = cust_email.indexOf(";",pos+1);
        }
        String email[] = new String[num];
        int start = 1;
        int end = cust_email.indexOf(";");

        for(int i = 0 ; i<num -1  ; i++){
            email[i] = cust_email.substring(start+1,end);
            start = end;
            end = cust_email.indexOf(";",end+1);

        }
        email[num-1]=cust_email.substring(start+1);
        return email;
    }

    public void setTenderFields(){
        if(!articleType.equals("2")){//άρθρο
            tenderExpireDate = "null";
            tenderAmount = "null";
            foreas = "null";
        }
        else{//δημοπρασία
            setTenderExpireDate() ;
            setTenderAmount() ;
            headline = "null";
        }
    }

    private void setTenderExpireDate() {
        try{
           int pos = headline.indexOf("/");
           if(pos!= -1){
               //get the date portions
            tenderExpireDate =
headline.substring(pos+1,headline.length());
            int pos2 = tenderExpireDate.indexOf("-");
            //change the format from DD-MM-YYYY to YYYY-MM-DD
            if(pos2 != -1){
                String DD = tenderExpireDate.substring(0,pos2);
                String MM =
tenderExpireDate.substring(pos2+1,pos2+3);
                String YYYY =
tenderExpireDate.substring(pos2+4,pos2+8);
                tenderExpireDate = YYYY + "-" + MM + "-" + DD;
            }
            else{
                if(tenderExpireDate.equals("0")){
                    tenderExpireDate = "null";
                }
                else{
                    throw new Exception("Error on
setTenderExpireDate() methode");
                }
            }
           }
           else {
               throw new Exception("Error on setTenderExpireDate()
methode. Character '/' not found in headline.");
           }
        }
```

```
        catch(Exception e){
          // σε πολλά XML, το format "<amound>/<DD-MM-YYY>", δεν
είναι καταχωρημένο σωστά και χτυπάει αυτό
          //το exception. Ως αποτλέλεσμα το query που σχηματίζεται
για την βάση δεν είναι σωστό
          //και δεν γίνεται το update της βάσης.
            tenderExpireDate = null;
        }

    }

    private void setTenderAmount() {
        try{
            if(headline.indexOf("/")!= -1){
                this.tenderAmount = headline.substring(0,
headline.indexOf("/"));
                if(this.tenderAmount.equals("0")) {
                    this.tenderAmount = "null";
                }
            }
            else {
               throw new Exception("Error on setTenderExpireDate()
methode. Character '/' not found in headline.");
             }
        }catch(Exception e){
          // σε πολλά XML, το format "<amound>/<DD-MM-YYY>", δεν
είναι καταχωρημένο σωστά και χτυπάει αυτό
          //το exception. Ως αποτέλεσμα το query που σχηματίζεται για
την βάση δεν είναι σωστό
          //και δεν γίνεται το update της βάσης.
            tenderAmount = null;
        }

    }

}
```

PrepareFile.java

```
package pdf2djvu;

import java.io.*;

public class PrepareFile {

    public String forParsing(String path) throws
FileNotFoundException, UnsupportedEncodingException, IOException{

        //άνοιγμα του xmlFilename.xml για διάβασμα. Είναι σε UTF-16
encoding
        FileInputStream fis = new FileInputStream(path);
        InputStreamReader in = new InputStreamReader(fis, "UTF-16");
        StringBuffer text = new StringBuffer();

        //δημιουργία αρχείου xmlFilename.xml_tmp με βάση το default
encoding
        String tmpFile = path.substring(0,path.length()-
4)+".xml_tmp";
        BufferedWriter bw = new BufferedWriter(new
FileWriter(tmpFile));

         int charac = in.read();
        while(charac != -1){
             text.append((char)charac);
           charac = in.read();
        }
         String strText = text.toString();

         //έλεγχος του <articleText> για ειδικούς χαρακτήρες.
         int start = strText.indexOf("<articleText>");
         int end = strText.indexOf("</articleText>");
         String ArticleTextField = strText.substring(start+13,end);//
String tag = "<articleText>"; int length = tag.length(); --> το
length είναι 13.

         String ArticleTextFieldwithTags =
strText.substring(start,end+14);
         strText =
strText.replace(ArticleTextFieldwithTags,ArticleTextFieldwithTags.rep
lace(ArticleTextField,""));
         if(!ArticleTextField.contains("&amp;")){
            ArticleTextField =
ArticleTextField.replaceAll("&","&amp;");
         }
         if(!ArticleTextField.contains("&lt;")){
            ArticleTextField =
ArticleTextField.replaceAll("<","&lt;");
         }
         if(!ArticleTextField.contains("&gt;")){
            ArticleTextField =
ArticleTextField.replaceAll(">","&gt;");
         }
         if(!ArticleTextField.contains("&quot;")){
            ArticleTextField =
ArticleTextField.replaceAll("“","&quot;");
```

```
         }
         if(!ArticleTextField.contains("&#39")){
            ArticleTextField =
ArticleTextField.replaceAll("‘","&#39");
         }

         start = strText.indexOf("<articleText>");
         end = strText.indexOf("</articleText>");
         strText = strText.substring(0,start+13) + ArticleTextField +
strText.substring(end,strText.length());

         //έλεγχος του <headline> για ειδικούς χαρακτήρες
         int headlineStart = strText.indexOf("<headline>");
         int headlineEnd = strText.indexOf("</headline>");
         String HeadlineTextField = strText.substring(headlineStart +
10,headlineEnd);
         String HeadlineTextFieldwithTags =
strText.substring(headlineStart,headlineEnd + 11);
         strText = strText.replace
(HeadlineTextFieldwithTags,HeadlineTextFieldwithTags.replace(Headline
TextField, ""));
         //some times the headline is well formated and sometimes
not.
         if(!HeadlineTextField.contains("&amp;")){
             HeadlineTextField =
HeadlineTextField.replaceAll("&","&amp;");
         }
         if(!HeadlineTextField.contains("&lt;")){
            HeadlineTextField =
HeadlineTextField.replaceAll("<","&lt;");
         }
         if(!HeadlineTextField.contains("&gt;")){
            HeadlineTextField =
HeadlineTextField.replaceAll(">","&gt;");
         }
         if(!HeadlineTextField.contains("&quot;")){
            HeadlineTextField =
HeadlineTextField.replaceAll("\"","&quot;");
         }
         if(!HeadlineTextField.contains("&apos;")){
            HeadlineTextField =
HeadlineTextField.replaceAll("'","&apos;");
         }

         headlineStart = strText.indexOf("<headline>");
         headlineEnd = strText.indexOf("</headline>");
         strText = strText.substring(0,headlineStart+10) +
HeadlineTextField + strText.substring(headlineEnd,strText.length());

         bw.write(strText);
            in.close();
            bw.close();
           //δημιουργία xmlFilename.xml_tmp2 αρχείου σε UTF-16
encoding
            String finalFilePath =tmpFile + "2";
            File finalFileF = new File(finalFilePath);
            finalFileF.createNewFile();
            BufferedReader br = new BufferedReader(new
FileReader(tmpFile));
            OutputStreamWriter osw = new OutputStreamWriter(new
FileOutputStream(finalFileF),"UTF-16");
```

```
            String s = br.readLine();
            while(s!=null){
                osw.write(s);
                s = br.readLine();
            }

            br.close();
            osw.close();

            File t = new File(tmpFile);
            while(t.exists()){
                t.delete();
            }

            return finalFileF.toString();
    }

}
```

DistributionContentHandler.java

```
package pdf2djvu;

import org.xml.sax.*;


public class DistributionContentHandler implements ContentHandler {
DistributionXmlObj DistrXmlObj ;

public DistributionXmlObj getDistrXmlObj() {
    return DistrXmlObj;
}

String str = null;

  public void characters(char[] text, int start, int length)
      throws SAXException {
        str = String.valueOf(text,start,length);
    }

  public void startElement(String namespaceURI, String localName,
   String qualifiedName, Attributes atts) {
      if(qualifiedName.equals("clip")){
        DistrXmlObj = new DistributionXmlObj();
      }
      if(qualifiedName.equals("author")){
        str = null;
       }
      if(qualifiedName.equals("id")){
        str = null;
      }
      if(qualifiedName.equals("artId")){
        str = null;
      }
      if(qualifiedName.equals("headline")){
        str = null;
      }
      if(qualifiedName.equals("pageName")){
        str = null;
      }
      if(qualifiedName.equals("numberOfClippings")){
        str = null;
      }
      if(qualifiedName.equals("articleSize")){
        str = null;
      }
      if(qualifiedName.equals("articleType")){
        str = null;
      }
      if(qualifiedName.equals("foreas")){
        str = null;
      }
      if(qualifiedName.equals("pageSize")){
        str = null;
      }
      if(qualifiedName.equals("publicationDate")){
        str = null;
```

```
    }
    if(qualifiedName.equals("timeCreated")){
      str = null;
    }
    if(qualifiedName.equals("searchWord")){
      str = null;
    }
    if(qualifiedName.equals("searchWordId")){
      str = null;
    }
    if(qualifiedName.equals("publicationId")){
      str = null;
    }
    if(qualifiedName.equals("publicationName")){
      str = null;
    }
    if(qualifiedName.equals("publicationCirc")){
      str = null;
    }
    if(qualifiedName.equals("paper_type")){
      str = null;
    }
    if(qualifiedName.equals("customerId")){
      str = null;
    }
    if(qualifiedName.equals("customerName")){
      str = null;
    }
    if(qualifiedName.equals("cust_resp")){
      str = null;
    }
    if(qualifiedName.equals("cust_city")){
      str = null;
    }
    if(qualifiedName.equals("cust_email")){
      str = null;
    }
    if(qualifiedName.equals("articleText")){
      str = null;
    }
}


 public void endElement(String namespaceURI, String localName,
 String qualifiedName) {
    if(qualifiedName.equals("author")){
     DistrXmlObj.setAuthor(checkString(str));
    }
    if(qualifiedName.equals("id")){
        DistrXmlObj.setId(str);
    }
    if(qualifiedName.equals("artId")){
        DistrXmlObj.setArtId(str);
    }
    if(qualifiedName.equals("headline")){
        DistrXmlObj.setHeadline(checkString(str));
    }
    if(qualifiedName.equals("pageName")){
        DistrXmlObj.setPageName(str);
    }
    if(qualifiedName.equals("articleSize")){
```

```
     DistrXmlObj.setArticleSize(str);
}
if(qualifiedName.equals("articleType")){
    DistrXmlObj.setArticleType(str);
}
if(qualifiedName.equals("foreas")){
     DistrXmlObj.setForeas(checkString(str));
}
if(qualifiedName.equals("numberOfClippings")){
     DistrXmlObj.setNumberOfClippings(str);
}
if(qualifiedName.equals("pageSize")){
    DistrXmlObj.setPageSize(str);
}
if(qualifiedName.equals("publicationDate")){
   DistrXmlObj.setPublicationDate(str);

}
if(qualifiedName.equals("timeCreated")){
   DistrXmlObj.setTimeCreated(str);

}
if(qualifiedName.equals("searchWord")){
   DistrXmlObj.setSearchWord(checkString(str));

}
if(qualifiedName.equals("searchWordId")){
   DistrXmlObj.setSearchWordId(str);

}
if(qualifiedName.equals("publicationId")){
    DistrXmlObj.setPublicationId(str);
}
if(qualifiedName.equals("publicationName")){
   DistrXmlObj.setPublicationName(checkString(str));

}
if(qualifiedName.equals("publicationCirc")){
   DistrXmlObj.setPublicationCirc(str);

}
if(qualifiedName.equals("paper_type")){
   DistrXmlObj.setPaper_type(str);

}

if(qualifiedName.equals("customerId")){
   DistrXmlObj.setCustomerId(str);

}
if(qualifiedName.equals("customerName")){
   DistrXmlObj.setCustomerName(checkString(str));

}

if(qualifiedName.equals("cust_resp")){
    DistrXmlObj.setCust_resp(checkString(str));

}
if(qualifiedName.equals("cust_city")){
    DistrXmlObj.setCust_city(checkString(str));
```

```
        }
        if(qualifiedName.equals("cust_email")){
           if(str == null) return;
           //υπάρχει περίπτωση στο πεδίο του email να υπάρχουν πολλαπλά
email. Το email θα στέλνετε μόνο στη πρώτη διεύθυνση.
           if(str.indexOf(";") != -1){
               str = str.substring(0,str.indexOf(";"));
           }
           DistrXmlObj.setCust_email(str.trim());
        }

        if(qualifiedName.equals("articleText")){
            DistrXmlObj.setArticleText(checkString(str));
        }
        if(qualifiedName.equals("clip")){
            DistrXmlObj.setTenderFields();
        }

    }

    // do-nothing methods
    public void setDocumentLocator(Locator locator) {}
    public void startDocument() {}
    public void endDocument() {}
    public void startPrefixMapping(String prefix, String uri) {}
    public void endPrefixMapping(String prefix) {}
    public void ignorableWhitespace(char[] text, int start, int length)
throws SAXException {}
    public void processingInstruction(String target, String data){}
    public void skippedEntity(String name) {
    }

    //επαναφορά των ειδικών χαρακτήρων.
    private String checkString(String str){
        if(str == null) return null;
        String tmp = str.replaceAll("&apos;", "'");
         tmp = tmp.replaceAll("&amp;", "&");
         tmp = tmp.replaceAll("&lt;", "<");
         tmp = tmp.replaceAll("&gt;", ">");
         tmp = tmp.replaceAll("&quot;", "\"");

        return tmp;
    }

    public String toString(){
        return DistrXmlObj.toString();

    }
}
```

ReadDistributionXML.java

```
package pdf2djvu;

import java.io.File;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.io.UnsupportedEncodingException;
import org.xml.sax.InputSource;
import org.xml.sax.XMLReader;
import org.xml.sax.helpers.XMLReaderFactory;

public class ReadDistributionXML{
    private String source;
    private DistributionContentHandler distContentHandl;


    public ReadDistributionXML(String filePath) throws
FileNotFoundException, UnsupportedEncodingException, IOException{
        super();
        PrepareFile pf = new PrepareFile();
        String PathOfxml_WithNoSpecialChars =
pf.forParsing(filePath);
        source = "file:///"+PathOfxml_WithNoSpecialChars;

        try{
           XMLReader parser = XMLReaderFactory.createXMLReader();
           distContentHandl = new DistributionContentHandler();
           parser.setContentHandler(distContentHandl);
           parser.parse(new InputSource(source));

           //διαγραφή του xmlFilename.xml_tmp2
           File f = new File(PathOfxml_WithNoSpecialChars);
           while(f.exists()){
            f.delete();
           }
        }
        catch(org.xml.sax.SAXException e){
            System.out.println("The "+source+" is not well formed.");
            System.out.println(e.getMessage());
            System.out.println(e.getStackTrace());
        }
        catch(IOException e){
            System.out.println(e.getMessage());
        }catch(Exception e){
            System.out.println(e.getMessage());
        }
    }

    public DistributionContentHandler getDistContentHandl() {
        return distContentHandl;
    }
    public DistributionXmlObj getDistributionXmlObj(){
        return distContentHandl.getDistrXmlObj();
    }

}
```

LastModifiedFileComparator.java

```
package pdf2djvu;


import java.io.File;
import java.io.Serializable;
import java.util.Comparator;

public class LastModifiedFileComparator implements Comparator,
Serializable {

    public static final Comparator LASTMODIFIED_COMPARATOR = new
LastModifiedFileComparator();


    /**
     * Compare the last the last modified date/time of two files.
     *
     * @param obj1 The first file to compare
     * @param obj2 The second file to compare
     * @return a negative value if the first file's lastmodified
date/time
     * is less than the second, zero if the lastmodified date/time
are the
     * same and a positive value if the first files lastmodified
date/time
     * is greater than the second file.
     *
     */
    public int compare(Object obj1, Object obj2) {
        File file1 = new File((String)obj1);
        File file2 = new File((String)obj2);
        long result = file1.lastModified() - file2.lastModified();
        if (result < 0) {
            return -1;
        } else if (result > 0) {
            return 1;
        } else {
            return 0;
        }
    }
}
```

PdfExtensionFilter.java

```
package pdf2djvu;

import java.io.File;
import java.io.FilenameFilter;
import javax.swing.filechooser.FileNameExtensionFilter;


public class PdfExtensionFilter implements FilenameFilter {

    FileNameExtensionFilter ff;

     /**
     * Δημιουργεί ένα "PDF file filter" φίλτρο το οποίο θα επιστρέψει
     * μόνο τα ονόματα των αρχείων που έχουν κατάληξη ".PDF" ή ".pdf"
     */
    public PdfExtensionFilter() {
        ff = new FileNameExtensionFilter("PDF file
filter","PDF","pdf");
    }

    public boolean accept(File dir, String name) {
        if(ff.accept(new File(name))) return true;
        return false;
    }

}
```

CheckAndGetSortedList.java

```
package pdf2djvu;

import java.io.File;
import java.io.FilenameFilter;
import java.util.Arrays;

public class CheckAndGetSortedList {

    static File rootPath;

     /**
     * Επιστρέφει ένα πίνακα με τα περιεχόμενα ενός φακέλου
     * τα οποία ικανοποιούν την συνθήκη του φίλτρου,
     * ταξινομημένα με βάση την ημερομηνία τροποποίησης (φθήνουσα
σειρά).
     * Τα περιεχόμενα του φακέλου τα οποία γίνονται δεκτά από το
φίλτρο
     * πρέπει απαραίτητα να έχουν αριθμητικό όνομα αρχείου.
     *
     * @param _rootPath Η διεύθυνση για την οποία θέλουμε
     * να πάρουμε τα περιεχόμενα.
     * @param fnf Το φίλτρο.
     * @return Τα ονόματα των αρχείων (χωρίς την κατάληξη)
     * καταχωρούνται σε ένα πίνακα ακεραίων και επιστρέφονται από την
μέθοδο.
     * Επιστρέφει null αν δεν υπάρχουν αρχεία στην διεύθυνση φακέλου
     * τα οποία να ικανοποιούν την συνθήκη του φίλτρου.
     */
    public  static Integer[] from(String _rootPath,FilenameFilter
fnf){
        String[] contents;
        rootPath = new File(_rootPath);
        contents = rootPath.list(fnf);
        if(contents.length>0 ){
            return orderByModifiedDate(_rootPath,contents);
        }
        else{
            return null;
        }
    }

     private  static Integer[] orderByModifiedDate(String rootPath,
String[] fileNames){
         Integer[] tmp = new Integer[fileNames.length];
         for(int i=0 ; i<fileNames.length; i++){
             fileNames[i]= rootPath + "\\" +fileNames[i];
         }


Arrays.sort(fileNames,LastModifiedFileComparator.LASTMODIFIED_COMPARA
TOR);

         for(int i=0; i<fileNames.length;i++){
             fileNames[i]=fileNames[i].replace(rootPath+"\\", "");
```

```
                tmp[i]= new
Integer(Integer.parseInt(fileNames[i].substring(0,fileNames[i].length
()-4)));
            }
            return tmp;

        }

}
```

StdOutErrLevel.java

```
package pdf2djvu;

import java.io.InvalidObjectException;
import java.io.ObjectStreamException;
import java.util.logging.Level;

/**
 * Class defining 2 new Logging levels, one for STDOUT, one for
STDERR,
 * used when multiplexing STDOUT and STDERR into the same rolling log
file
 * via the Java Logging APIs.
 */
public class StdOutErrLevel extends Level {

    /**
     * Private constructor
     */
    private StdOutErrLevel(String name, int value) {
        super(name, value);
    }
    /**
     * Level for STDOUT activity.
     */
    public static Level STDOUT =
        new StdOutErrLevel("STDOUT", Level.INFO.intValue()+53);
    /**
     * Level for STDERR activity
     */
    public static Level STDERR =
        new StdOutErrLevel("STDERR", Level.INFO.intValue()+54);
    /**
     * Method to avoid creating duplicate instances when
deserializing the
     * object.
     * @return the singleton instance of this <code>Level</code>
value in this
     * classloader
     * @throws ObjectStreamException If unable to deserialize
     */
    protected Object readResolve()
        throws ObjectStreamException {
        if (this.intValue() == STDOUT.intValue())
            return STDOUT;

        throw new InvalidObjectException("Unknown instance :" +
this);
    }

}
```

LoggingOutputStream.java

```
package pdf2djvu;

import java.io.ByteArrayOutputStream;
import java.io.IOException;
import java.util.logging.Level;
import java.util.logging.Logger;

/**
 * An OutputStream that writes contents to a Logger upon each call to
flush()
 */
class LoggingOutputStream extends ByteArrayOutputStream {

    private String lineSeparator;

    private Logger logger;
    private Level level;

    /**
     * Constructor
     * @param logger Logger to write to
     * @param level Level at which to write the log message
     */
    public LoggingOutputStream(Logger logger, Level level) {
        super();
        this.logger = logger;
        this.level = level;
        lineSeparator = System.getProperty("line.separator");
    }

    /**
     * upon flush() write the existing contents of the OutputStream
     * to the logger as a log record.
     * @throws java.io.IOException in case of error
     */
    public void flush() throws IOException {

        String record;
        synchronized(this) {
            super.flush();
            record = this.toString();
            super.reset();

            if (record.length() == 0 || record.equals(lineSeparator))
{
                // avoid empty records
                return;
            }

            logger.logp(level, "", "", record);
        }
    }
}
```

LoggingInitialization.java

```
package pdf2djvu;

import java.io.PrintStream;
import java.util.logging.FileHandler;
import java.util.logging.Handler;
//import java.util.logging.LogManager;
import java.util.logging.Logger;
import java.util.logging.SimpleFormatter;

public class LoggingInitialization {
    public static void Initialize(String logFile, int sizeInMB, int
NumberOfFiles,boolean toAppend) throws Exception {

        // initialize logging to go to rolling log file
        //There is a global LogManager object that keeps track of
global logging information.
//        LogManager logManager = LogManager.getLogManager();
//        logManager.reset();

        // log file max size 1M, 3 rolling files, append-on-open
        //FileHandler: A handler that writes formatted log records
either to a single file, or to a set of rotating log files.
        Handler fileHandler = new FileHandler(logFile, sizeInMB *
1000000, NumberOfFiles, toAppend);
        //SimpleFormatter: Writes brief "human-readable" summaries of
log records.
        fileHandler.setFormatter(new SimpleFormatter());
        //  The root Logger (named "") has no parent. Anonymous
loggers are all given the root logger as their parent.
        Logger.getLogger("").addHandler(fileHandler);



        // preserve old stdout/stderr streams in case they might be
useful
        PrintStream stdout = System.out;
        PrintStream stderr = System.err;

        // now rebind stdout/stderr to logger

        Logger logout = Logger.getLogger("stdout");
        Logger loger = Logger.getLogger("stderr");
        LoggingOutputStream losout = new LoggingOutputStream(logout,
StdOutErrLevel.STDOUT);
        LoggingOutputStream loserr = new LoggingOutputStream(loger,
StdOutErrLevel.STDERR);
        System.setOut(new PrintStream(losout, true));
        System.setErr(new PrintStream(loserr, true));

    }

}
```

Mail.java

```
package pdf2djvu;

import java.util.Properties;
import javax.mail.*;
import javax.mail.internet.*;


public class Mail {

 Properties props = null;
 MimeMessage message = null;
 EmailFields ef = null;

public Mail(EmailFields m){
    ef = m;
    props = new Properties();
}


public  void sendMail() {
     // Setup mail server
    props.put("mail.smtp.host", ef.getHost());

    // Get session
    Session session = Session.getDefaultInstance(props, null);
    try{
    // Define message
    message = new MimeMessage(session);
    message.setSubject(ef.getSubject());
    String to[] = ef.getTo();
     for(int i = 0; i<to.length ; i++){
        if(to[i] != null){
             message.addRecipient(Message.RecipientType.TO, new
InternetAddress(to[i]));
        }
    }
    message.setFrom(new InternetAddress(ef.getFrom()));
    message.setText(ef.getBody());

    // Send message
    Transport.send(message);
    System.out.println("Email send successfully.");
    }
    catch(MessagingException e){
        System.out.println("MessagingException:");
        System.out.println(e.getMessage());
    }
    catch(Exception exc){
        System.out.println("Sending message - Other exception: " +
exc.getMessage());
    }
 }

}
```

EmailFields.java

```
package pdf2djvu;

public class EmailFields {
  private String host;
  private String from;
  private String to[]= new String[15];
  private String subject;
  private String body_sample_file = "";
  private String body = "";

    public EmailFields() {
    }
    public EmailFields(EmailFields ef) {
       host = ef.host;
       from = ef.from;
       to = ef.to;
       subject = ef.subject;
       body_sample_file = ef.body_sample_file;
       body = ef.body;
    }

  public String getHost(){
    return host;
  }
  public String getFrom(){
     return from;
  }
  public String[] getTo(){
      return to;
  }
  public String getSubject(){
    return subject;
  }
  public String getBody_sample_file(){
    return body_sample_file;
  }

  public String getBody() {
    return body;
  }

  public void setHost(String str){
   host = str;
  }
  public void setFrom(String str){
   from = str;
  }
  public void setTo(String[] str){
      to= str;
  }
  public void setTo(String str){
      to[0]= str;
  }
  public void setSubject(String str){
    subject = str;
  }
```

```
  public void setBody_sample_file(String str){
    body_sample_file = str;
  }

  //Προαιρετικά, στην περίπτωση που δεν χρησιμοποιείται
//  κάποιο sample αρχείο για body, ορίζει ποιό θα είναι το body του
email
    public void setBody(String body) {
        this.body = body;
    }

}
```

NewClientEmail.java

```
package pdf2djvu;

import java.io.BufferedReader;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;

public class NewClientEmail extends EmailFields{
  private String body ;

    public NewClientEmail(EmailFields tmp, String
customerRecipient,int number) {
        super(tmp);
        String[] otherRecipients = tmp.getTo();
        // το "otherRecipients" είναι κοινοποιήσεις.
        if(otherRecipients!=null){
            String recipients[] = new String[otherRecipients.length +
1];
            recipients[0] = customerRecipient;
            for (int i=0; i<otherRecipients.length; i++){
                recipients[i+1] = otherRecipients[i];
            }
            this.setTo(recipients);
        }else{
            this.setTo(customerRecipient);
        }

        try {
            edit_newClient_Body(this.getBody_sample_file(),number);
        } catch (FileNotFoundException ex) {
            System.out.println("FileNotFoundException");
            System.out.println(ex.getMessage());
        } catch (IOException ex) {
            System.out.println("IOException");
            System.out.println(ex.getMessage());
        }


    }

      private void edit_newClient_Body(String sampleFilePath, int
number) throws FileNotFoundException, IOException{

      File sample = new File(sampleFilePath);
      BufferedReader br = new BufferedReader(new FileReader(sample));
      String line = "";
      String text = line;
      while(line != null){

          if(line.contains("@") && line.length() == 1){
            line = "\r\nUser Name: "+ number +"\r\nPassword: " +
number;
            text += line +"\r\n";
          }
          else{
```

```
                text += line+"\r\n";
            }
            line = br.readLine();
        }//while
        body = text;
        br.close();
    }

        public String getBody(){
            return body;
        }
}
```

EmailContainer.java

```
package pdf2djvu;


public class EmailContainer {
    EmailFields emails[] ; //0 --> subj: Error converting pdf  1 -->
subj: new customer
    int current= 0;

    public EmailContainer() {
        emails = new EmailFields[15];
    }

    public void setEmail(EmailFields e){
        emails[current] = e;
        current ++;
    }


    public EmailFields getEmail(int i){
        return emails[i];
    }
    public int getSize(){
        int i=0;
        for(int j=0;j<emails.length;j++){
            if(emails[i]!=null) i++;
        }
        return i;
    }



}
```

Company.java

```
package pdf2djvu;

import java.util.Vector;


 public class Company{
       private String code;
       private String company_name;
       private String person_FullName;
       private String email;
       private String city;
       private Vector keywords;

       class Keyword{
           String name;
           int code;

        public Keyword(int code, String name) {
            this.name = name;
            this.code = code;
        }

        public int getCode() {
            return code;
        }

        public void setCode(int code) {
            this.code = code;
        }

        public String getName() {
            return name;
        }

        public void setName(String name) {
            this.name = name;
        }
       }


        public Company(String user_code, String company_name, String
person_FullName, String email, String city) {
            this.code = user_code;
            this.company_name = company_name;
            this.person_FullName = person_FullName;
            this.email = email;
            this.city = city;
            keywords = new Vector(1,1);
        }

        public int getNumberOfKeywords(){
            keywords.trimToSize();
            return keywords.size();
        }

        public void setKeyword(int code, String name){
```

```
            keywords.add(new Keyword(code,name));
        }

        public String getKeywordName(int i){
            return
String.valueOf(((Keyword)keywords.get(i)).getName());
        }

        public int getKeywordCode(int i){
            return
Integer.valueOf(((Keyword)keywords.get(i)).getCode());
        }

    public String getTown() {
        return city;
    }

    public void setCity(String city) {
        this.city = city;
    }

    public String getCompany_name() {
        return company_name;
    }

    public void setCompany_name(String company_name) {
        this.company_name = company_name;
    }

    public String getEmail() {
        return email;
    }

    public void setEmail(String email) {
        this.email = email;
    }

    public String getPerson_FullName() {
        return person_FullName;
    }

    public void setPerson_FullName(String person_FullName) {
        this.person_FullName = person_FullName;
    }

    public String getCode() {
        return code;
    }

    public void setCode(String user_code) {
        this.code = user_code;
    }
   }
```

ConnectAtOracleDB.java

```
package pdf2djvu;

import java.sql.*;
import java.util.Vector;


  class ConnectAtOracleDB {

    private ResultSet rs=null;

    //Το μέλος "hasResult" έχει την τιμή "true" αν η SQL έχει φέρει
τουλάχιστον ένα αποτέλεσμα.
    //Διαφορετικά, έχει την τιμή "false".
    private boolean hasResult=false;
    //Το μέλος "results" είναι ένας Vector ο οποίος περιέχει Vectors.
    //Ο κάθε Vector που εισάγεται στο results περιέχη τα δεδομένα
μίας εγγραφής που έχει επιστρέψει το ερώτημα
    //μετά την εκτέλεσή του.
    private Vector<Vector> results = new Vector<Vector>(1,1);
    //Το μέλος "columnsNames" είναι ένας Vector τα στοιχεία του οποία
είναι τα ονόματα
    // των πεδίων της βάσης.
    private Vector columnsNames= new Vector();
    //Το "rows_num" περιέχει τον αριθμό των εγγραφών που έχει φέρει η
εκτέλεση της SQL.
    private int rows_num ;

    private Connection dbconnection = null;
    private Statement s = null;

    private ConfigurationContentHandler contHand;

    public ConnectAtOracleDB(ConfigurationContentHandler contHand) {
        this.contHand = contHand;
    }


    public boolean Connect(){
      try {
      Class.forName("oracle.jdbc.driver.OracleDriver");

      java.util.Properties info = new java.util.Properties();
      info.put ("user", contHand.getOracleUsername());
      info.put ("password",contHand.getOraclePassword());
      info.put("internal_logon", contHand.getOracleInernalLogon());

dbconnection=DriverManager.getConnection("jdbc:oracle:thin:@"+contHan
d.getOracleIp()+":prod",info);

       }catch(Exception e){
            System.out.println("Error on connection!");
            System.out.println(e.getMessage());
            return false;
        }
     return true;
    }
```

```
    //Αυτή η μέθοδος εκτελεί την SQL εντολή (μόνο Select) που δίνεται
ως παράμετρος.
    //Στη συνέχεια, αποθηκεύει τα αποτελέσματα στις τοπικές
μεταβλητές.
    public void Execute(String SQL) throws SQLException{
        resetLocalResults();
        s = dbconnection.createStatement();
        results  = new Vector();
        if(SQL.charAt(0)=='S' || SQL.charAt(0)=='s'){
            rs= s.executeQuery(SQL);

            ResultSetMetaData rmd = rs.getMetaData();
            int col_num = rmd.getColumnCount();
            for (int i=0;i<col_num;i++){
                columnsNames.add(rmd.getColumnLabel(i+1));
            }

            Vector<String> row ;
            while(rs.next()){
                row = new Vector<String>(col_num);
                for(int m=0;m<col_num;m++){
                    row.add(rs.getString(m+1));
                }
                results .add(row);
                rows_num++;
            }
            results .trimToSize();
            if(rows_num > 0) hasResult = true;
            else hasResult = false;

        }
    }

    private void resetLocalResults(){
      hasResult=false;
      results = new Vector<Vector>();
      columnsNames= new Vector();
      rows_num = 0;
    }

    public boolean Close(){
        try{
            dbconnection.close();
            return true;
        }
        catch(Exception e){System.out.println(e.getMessage()); return
false;}
    }

    public Vector getColumnsNames() {
        return columnsNames;
    }

    public Vector<Vector> getResults() {
        return results ;
    }

    public Connection getDbconnection() {
        return dbconnection;
    }
```

```
    public boolean isHasResult() {
        return hasResult;
    }

    public int getRows_num() {
        return rows_num;
    }
}
```

OracleDatabaseQueries.java

Στο παρακάτω κώδικα παραλείπονται τα ερωτήματα (query) που απευθύνονται στην βάση του παραγωγικού υποσυστήματος για λόγους ασφάλειας. Η λογική αυτών όμως δίνεται στο κείμενο της πτυχιακής.

```
package pdf2djvu;

import java.sql.SQLException;
import java.util.Vector;

public class OracleDatabaseQueries {

    ConnectAtOracleDB connectionObj;

    public OracleDatabaseQueries(ConfigurationContentHandler
contHand) {
         connectionObj = new ConnectAtOracleDB(contHand);
    }

     public Vector<Company> getNewCustomers(String webCustStr) throws
SQLException, Exception{

        if(!connectionObj.Connect()) throw new Exception("Unable to
connect to Oracle Database. Check log file for Details.");
        try{
            //Ποιοί είναι οι πελάτες που πρέπει να δημιουργηούν
            String SQLgetActiveClients = "Select * from ( "+
            "select distinct customer_number, customer_name,
customer_email, customer_firstname, customer_lastname,
customer_address "+
            " from {…}"+
            " where {…} and"+
            " customer_number not in " + webCustStr +")";

            connectionObj.Execute(SQLgetActiveClients) ;
            Vector<Vector> results = connectionObj.getResults();
            if(!connectionObj.isHasResult()) return null;
            Vector<Company> newClients = new Vector(1,1);
            for(int k=0 ; k<connectionObj.getRows_num(); k++){
                Vector row = (Vector)results.get(k);
                newClients.add( new
Company(String.valueOf(row.get(0)), String.valueOf(row.get(1)),
String.valueOf(row.get(3)) + " " +
                        String.valueOf(row.get(4)),
String.valueOf(row.get(2)), String.valueOf(row.get(5))));
            }

            //Τα keyword συγκεκριμένου πελάτη
            for(int k=0 ; k<newClients.size(); k++){
                Company c = getCustomerKeywords(newClients.get(k));
                newClients.setElementAt(c,k);
            }
            newClients.trimToSize();
            return newClients;
        }finally{
            connectionObj.Close();
```

```
        }
        }

     private Company getCustomerKeywords(Company company) throws
SQLException{

         String SQLgetClientKeywords = "select distinct keyword_id,
keyword_name " +
                "from {…} " +
                "where {…} " +
                "and customer_number = '" + company.getCode()  + "'";
            connectionObj.Execute(SQLgetClientKeywords);
            if(!connectionObj.isHasResult()) return null;
            Vector<Vector> results = connectionObj.getResults();
            Vector row;
            for(int i=0; i<connectionObj.getRows_num(); i++){
                row = results.get(i);

company.setKeyword(Integer.valueOf(String.valueOf(row.get(0))),String
.valueOf(row.get(1)));
            }
            return company;

     }



    public Vector<Company> getActiveCustomersKeywords(String
webCustStr) throws SQLException, Exception{

        if(!connectionObj.Connect()) throw new Exception("Unable to
connect to Oracle Database. Check log file for Details.");
        try{
        //Ποιοί είναι οι πελάτες που πρέπει να δημιουργηούν
            String SQLgetActiveClients = "Select * from ( "+
            "select distinct customer_number, customer_name,
customer_email, customer_firstname, customer_lastname,
customer_address "+
            " from {…}"+
            " where {…}"+
            " and customer_number in " + webCustStr +")";

            connectionObj.Execute(SQLgetActiveClients);
            if(!connectionObj.isHasResult()) return null;
            Vector results = connectionObj.getResults();
            Vector<Company> activeClients = new Vector<Company>(1,1);

            for(int k=0 ; k<connectionObj.getRows_num(); k++){
                Vector row = (Vector)results.get(k);
                activeClients.add( new
Company(String.valueOf(row.get(0)), String.valueOf(row.get(1)),
String.valueOf(row.get(3)) + " " +
                        String.valueOf(row.get(4)),
String.valueOf(row.get(2)), String.valueOf(row.get(5))));
            }

                //Τα keyword συγκεκριμένου πελάτη
            for(int k=0 ; k<activeClients.size(); k++){
                Company c =
getCustomerKeywords(activeClients.get(k));
                activeClients.setElementAt(c, k);
```

```
            }
            activeClients.trimToSize();
            return activeClients;
        }finally{
            connectionObj.Close();
        }
    }


    public String[] getCustomersToActivate(String webCustStr) throws
SQLException, Exception{
        if(!connectionObj.Connect()) throw new Exception("Unable to
connect to Oracle Database. Check log file for Details.");
        try{
            //Ποίοι είναι οι πελάτες που πρέπει να γίνουν ενεργοί
            String SQLgetActiveClients = "Select * from ( "+
            "select distinct customer_number "+
            " from {…}"+
            " where {…}"+
            " and customer_number " + webCustStr +")";

            connectionObj.Execute(SQLgetActiveClients);
            if(!connectionObj.isHasResult()) return null;
            Vector results = connectionObj.getResults();
            String custToActivate[] = new
String[connectionObj.getRows_num()];
            for(int k=0 ; k<connectionObj.getRows_num(); k++){
                Vector row = (Vector)results.get(k);
                custToActivate[k] = String.valueOf(row.get(0));
            }
            return custToActivate;
        }finally{
            connectionObj.Close();
        }
    }

    public String[] getCustomersToDeactiv(String webCustStr) throws
SQLException, Exception{
        if(!connectionObj.Connect()) throw new Exception("Unable to
connect to Oracle Database. Check log file for Details.");
        try{
            //Ποιοί είναι οι πελάτες που πρέπει να απενεργοποιηθούν.
            String SQLgetInactiveClients = "Select * from ( "+
            "select distinct customer_number "+
            " from {…}"+
            " where {…} and"+
            " customer_number not in (select distinct customer_number
from {…} where {…} " +
            ") and" +
            " customer_number in " + webCustStr +")";

            connectionObj.Execute(SQLgetInactiveClients);
            if(!connectionObj.isHasResult()) return null;
            Vector results = connectionObj.getResults();
            String custToDeactivate[] = new
String[connectionObj.getRows_num()];
            for(int k=0 ; k<connectionObj.getRows_num(); k++){
                Vector row = (Vector)results.get(k);
                custToDeactivate[k] = String.valueOf(row.get(0));
            }
            return custToDeactivate;
```

```
        }finally{
            connectionObj.Close();
        }
    }//getCustomersToDeactiv()
}
```

ConnectAtSqlDB.java

```
package pdf2djvu;

import java.sql.*;
import java.util.Vector;
public class ConnectAtSqlDB {

    private Connection dbConnection = null;
    private Statement stmt = null;
    private ResultSet rs=null;
    private String connectionString;


    ///Το μέλος "hasResult" έχει την τιμή "true" αν η SQL έχει φέρει
τουλάχιστον ένα αποτέλεσμα.
    //Διαφορετικά, έχει την τιμή "false".
    private boolean hasResult=false;
    //Το μέλος "results" είναι ένας Vector ο οποίος περιέχει Vectors.
    //Ο κάθε Vector που εισάγεται στο results περιέχη τα δεδομένα
μίας εγγραφής που έχει επιστρέψει το ερώτημα
    //μετά την εκτέλεσή του.
    private Vector<Vector> results= new Vector<Vector>();
    //Το μέλος "columnsNames" είναι ένας Vector τα στοιχεία του οποία
είναι τα ονόματα
    // των πεδίων της βάσης.
    private Vector columnsNames= new Vector();
    //Το "rows_num" περιέχει τον αριθμό των εγγραφών που έχει φέρει η
εκτέλεση της SQL.
    private int rows_num ;
    //Στην περίπτωση εκτέλεσης οποιασδήποτε εντολής πλίν της Select,
το μέλος "afectedResults"
    //επιστρέφει το πλήθος των γραμμών που επιρεάστηκαν από το
ερώτημα SQL.
    private int afectedResults;


    public ConnectAtSqlDB(String ip, String database, String
username, String password) {
        connectionString =
"jdbc:sqlserver://"+ip+";user="+username+";password="+password+";data
base="+database;
    }

   public boolean Connection() throws SQLException{
        try{

Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");
        }
        catch(Exception e){
            System.out.println("The driver is not loaded!");
            System.out.println(e.getMessage());
            return false;
        }
       dbConnection = DriverManager.getConnection(connectionString);
        return true;
    }
```

```
    //Εκτελή την εντολή SQL που δίνεται ως παράμετρος. Το SQL μπορεί
να είναι οτιδήποτε.
    //(πχ Select, Update, Delete )
    public void Execute(String SQL) throws SQLException{
        resetLocalResults();
        stmt = dbConnection.createStatement();
        if(SQL.charAt(0)=='S' || SQL.charAt(0)=='s'){
            rs= stmt.executeQuery(SQL);
            ResultSetMetaData rmd = rs.getMetaData();
            int col_num = rmd.getColumnCount();
            for (int i=0;i<col_num;i++){
                Vector val = new Vector();
                val.add(rmd.getColumnLabel(i+1));
                columnsNames.add(val);
            }

            while(rs.next()){
                hasResult = true;
                Vector row = new Vector();
                for(int m=0;m<col_num;m++){
                    row.add(rs.getObject(m +1));
                }
                results.add(row);
                rows_num++;
            }
            if(rows_num > 0) hasResult = true;
            else hasResult = false;

        }
        else if(SQL.charAt(0)=='I' || SQL.charAt(0)=='i' ||
SQL.charAt(0)=='U' || SQL.charAt(0)=='u' || SQL.charAt(0)=='D' ||
SQL.charAt(0)=='d'){
            afectedResults= stmt.executeUpdate(SQL);
            if(afectedResults !=0) {
                hasResult=true;
            }
            else {
                hasResult = false;
            }
        }
    }

    private void resetLocalResults(){
        hasResult=false;
        columnsNames= new Vector();
        rows_num = 0;
        afectedResults = 0;
    }

   public void Close(){
        try{
        dbConnection.close();
        }
        catch(SQLException e){
            System.out.println(e.getMessage());
        }
    }

    public int getAfectedResults() {
        return afectedResults;
    }
```

```
    public Vector getColumnsNames() {
        return columnsNames;
    }

    public Connection getDbConnection() {
        return dbConnection;
    }

    public boolean isHasResult() {
        return hasResult;
    }

    public Vector<Vector> getResults() {
        return results;
    }

    public int getRows_num() {
        return rows_num;
    }
}
```

SQLDatabaseQueries.java

```
package pdf2djvu;

import java.sql.CallableStatement;
import java.sql.SQLException;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.Random;
import java.util.Vector;

public class SQLDatabaseQueries {


   private ConfigurationContentHandler contHand;
   private ConnectAtSqlDB connectionObj;
   public SQLDatabaseQueries(ConfigurationContentHandler contHand) {
            this.contHand = contHand;
            connectionObj = new
ConnectAtSqlDB(contHand.getSQLip(),contHand.getSQLdatabaseName(),cont
Hand.getSQLusername(),contHand.getSQLpassword());
    }
   Random rnd = new Random();
   int initialPassAndUsername = rnd.nextInt(5000000) + 1000000;

   //επιστρέφει κενό πίνακα αν δεν υπάρχουν πελάτες
   private String[] getAllCustomers() throws SQLException{
    String SQLCust = "Select code FROM Company;";
    String[] results = null;
    try{
      if(connectionObj.Connection()){
              connectionObj.Execute(SQLCust);
              if(!connectionObj.isHasResult()) return null;
              results= new String[connectionObj.getRows_num()];
              for(int j=0; j<results.length; j++){
                  Vector row = connectionObj.getResults().get(j);
                  results[j] = String.valueOf(row.get(0));
              }
         }
       return results;
       }finally{
           connectionObj.Close();
       }
  }

   public String getAllCustomersStr() throws SQLException{
       String results[] = getAllCustomers();
       return formatSQLsBracket(results);
   }

   public boolean createAccount(Company newClient,boolean sendEmail)
throws SQLException, Exception{

      CallableStatement proc_stmt = null;
      boolean returnValue = true;
      try{
          if(!connectionObj.Connection())  {
```

```
                throw new Exception("Unable to connect to SQL Database.
Check log file for Details.");

            }
            proc_stmt = connectionObj.getDbConnection().prepareCall("{
call sp_insertKeyword(?,?,?,?,?,?,?,?,?) }");

            for(int i=0; i<newClient.getNumberOfKeywords(); i++){
                proc_stmt.setString(1, newClient.getCode());
                proc_stmt.setString(2, newClient.getCompany_name());
                proc_stmt.setInt(3, newClient.getKeywordCode(i));
                proc_stmt.setString(4, newClient.getKeywordName(i));
                proc_stmt.setString(5, newClient.getPerson_FullName());
                proc_stmt.setString(6, newClient.getEmail());
                proc_stmt.setString(7, newClient.getTown());
                proc_stmt.registerOutParameter(8,
java.sql.Types.VARCHAR);
                proc_stmt.registerOutParameter(9,
java.sql.Types.INTEGER);
                try{
                  proc_stmt.execute();
                }
                finally{
                  int result = proc_stmt.getInt(9);
                  if(!(result==0)){
                      String message = "Error inserting keyword " +
newClient.getKeywordName(i) + " for account : " +
newClient.getCode()+ ". Error " + proc_stmt.getString(8);
                      System.out.println(message);
                      returnValue = false;
                  }
              }
            }

          if(sendEmail) {
             System.out.println("Send email to the mew client.");
              sendEmail2newClient(newClient.getEmail());
          }
          if(!returnValue){
              sendEmail2admin(contHand,newClient);
          }
          return returnValue;
       }
       finally{
          proc_stmt.close();
          connectionObj.Close();
       }
    }


    private String[] getStoppedCustomers() throws SQLException,
Exception{
     String SQLCust = "SELECT code FROM Company WHERE
contract_expiration_date IS NOT NULL;" ;
     String[] results = null;

      try{
           if(!connectionObj.Connection()) throw new Exception("Unable
to connect to SQL Database. Check log file for Details.");
                  connectionObj.Execute(SQLCust);
                  if(!connectionObj.isHasResult()) return null;
```

```
                results= new String[connectionObj.getRows_num()];
                Vector row;
                for(int j=0; j<results.length; j++){
                    row = connectionObj.getResults().get(j);
                    results[j] = String.valueOf(row.get(0));
                }

       return results;
      }finally{
        connectionObj.Close();
     }
   }

   public String getStoppedCustomersStr() throws SQLException,
Exception{
       String results[] = getStoppedCustomers();
       return formatSQLsBracket(results);
   }

   private String[] getActiveCustomers() throws SQLException,
Exception{
    String SQLCust = "SELECT code FROM Company WHERE
contract_expiration_date IS NULL;" ;
    String[] results = null;
      try{
          if(!connectionObj.Connection()) throw new Exception("Unable
to connect to SQL Database. Check log file for Details.");
                connectionObj.Execute(SQLCust);
                if(!connectionObj.isHasResult()) return null;
                results= new String[connectionObj.getRows_num()];
                for(int j=0; j<results.length; j++){
                    Vector row = connectionObj.getResults().get(j);
                    results[j] = String.valueOf(row.get(0));
                }

          return results;
      }finally{
          connectionObj.Close();
      }
   }

   public String getActiveCustomersStr() throws SQLException,
Exception{
       String results[] = getActiveCustomers();
       return formatSQLsBracket(results);
   }

   private String formatSQLsBracket(String[] results){
       if(results != null){
           String res = "(";
           for (int i = 0 ; i < results.length ; i++){
               res += "'"+results[i]+"'";
               if(i+1 == results.length){
                   res += ")";
               }else{
                   res += ",";
               }
           }
           return res;
       }else return null;
   }
```

```
   public boolean activateCompany(String code) throws SQLException{
       CallableStatement proc_stmt = null;
       try{
       if(connectionObj.Connection()){
            proc_stmt =
connectionObj.getDbConnection().prepareCall("{ call
sp_activateCompany(?,?,?) }");
            proc_stmt.setString(1,code);
            proc_stmt.registerOutParameter(2,
java.sql.Types.VARCHAR);
            proc_stmt.registerOutParameter(3,java.sql.Types.INTEGER);
            proc_stmt.execute();
        }
        }finally{
            int result = proc_stmt.getInt(3);
            if(!(result==0)){
                System.out.println("Error activating company : " +
code+ ". Error " + proc_stmt.getString(2));
                proc_stmt.close();
                connectionObj.Close();
                return false;
            }else{
             proc_stmt.close();
             connectionObj.Close();
             return true;
            }
       }
   }

   public boolean deactivateCompany(String code) throws SQLException
{

       CallableStatement proc_stmt = null;
       try{
        if(connectionObj.Connection()){
            proc_stmt =
connectionObj.getDbConnection().prepareCall("{ call
sp_deactivateCompany(?,?,?) }");
            proc_stmt.setString(1,code);
            proc_stmt.registerOutParameter(2,
java.sql.Types.VARCHAR);
            proc_stmt.registerOutParameter(3,java.sql.Types.INTEGER);
            proc_stmt.execute();
        }
        }finally{
            int result = proc_stmt.getInt(3);
            if(!(result==0)){
                System.out.println("Error deactivating company : " +
code+ ". Error " + proc_stmt.getString(2));
                proc_stmt.close();
                connectionObj.Close();
                return false;
            }else{
             proc_stmt.close();
             connectionObj.Close();
             return true;
            }
       }
   }
```

```
   public boolean insertClip(DistributionXmlObj xmlObj ) throws
SQLException{

        if(!xmlObj.getArticleType().equals("2")){
            try {
                java.sql.Date publicationDate =
java.sql.Date.valueOf(xmlObj.getPublicationDate());
                SimpleDateFormat format = new SimpleDateFormat("yyyy-
MM-dd'T'kk:mm:ss");
                java.util.Date date =
format.parse(xmlObj.getTimeCreated());
                java.sql.Timestamp dateCreated = new
java.sql.Timestamp(date.getTime());

                return
insertArticle(Integer.valueOf(xmlObj.getPublicationId()),xmlObj.getPu
blicationName(),  xmlObj.getPaper_type(),publicationDate,

Integer.valueOf(xmlObj.getPublicationCirc()),Integer.valueOf(xmlObj.g
etArtId()), xmlObj.getArticleText(),
xmlObj.getPageSize(),Integer.valueOf(xmlObj.getNumberOfClippings()),
                xmlObj.getArticleSize(),xmlObj.getPageName(),
                xmlObj.getHeadline(),xmlObj.getAuthor(),
                Integer.valueOf(xmlObj.getId()),dateCreated,
xmlObj.getCustomerId(),
xmlObj.getCustomerName(),Integer.valueOf(xmlObj.getSearchWordId()),
                xmlObj.getSearchWord(),xmlObj.getCust_resp(),
xmlObj.getCust_email(),String.valueOf(initialPassAndUsername),xmlObj.
getCust_city());

            } catch (ParseException ex) {
                System.out.println("Error parsing date : " +
xmlObj.getTimeCreated() + "(format: yyyy-MM-ddTkk:mm:ss) for article
id :"
                        + xmlObj.getArtId() + " in XML file: " +
xmlObj.getId() + ". Error: " + ex.getMessage()) ;
                return false;
            }

        }else{
            try {
                java.sql.Date publicationDate =
java.sql.Date.valueOf(xmlObj.getPublicationDate());
                SimpleDateFormat format = new SimpleDateFormat("yyyy-
MM-dd'T'kk:mm:ss");
                java.util.Date date =
format.parse(xmlObj.getTimeCreated());
                java.sql.Timestamp dateCreated = new
java.sql.Timestamp(date.getTime());

                java.sql.Date tenderExpireDate =
java.sql.Date.valueOf(xmlObj.getTenderExpireDate());

                return
insertTender(Integer.valueOf(xmlObj.getPublicationId()),xmlObj.getPub
licationName(),  xmlObj.getPaper_type(),publicationDate,

Integer.valueOf(xmlObj.getPublicationCirc()),Integer.valueOf(xmlObj.g
etArtId()), xmlObj.getArticleText(),
xmlObj.getPageSize(),Integer.valueOf(xmlObj.getNumberOfClippings()),
                xmlObj.getArticleSize(),xmlObj.getPageName(),
```

```
Integer.valueOf(xmlObj.getTenderAmount()),xmlObj.getForeas(),tenderEx
pireDate ,
                Integer.valueOf(xmlObj.getId()),dateCreated,
xmlObj.getCustomerId(),
xmlObj.getCustomerName(),Integer.valueOf(xmlObj.getSearchWordId()),

xmlObj.getSearchWord(),xmlObj.getCust_resp(),xmlObj.getCust_email(),S
tring.valueOf(initialPassAndUsername),xmlObj.getCust_city());
            } catch (ParseException ex) {
                System.out.println("Error parsing date : " +
xmlObj.getTimeCreated() + "(format: yyyy-MM-ddTkk:mm:ss) for article
id :"
                        + xmlObj.getArtId() + " in XML file: " +
xmlObj.getId() + ". Error: " + ex.getMessage()) ;
                return false;
            }
        }

    }

    private boolean insertArticle (int pressMedia_id, String
pressMedia_name,String pressMedia_type,
            java.sql.Date issue_publishedDate, int issue_copies, int
text_id , String text_ocr, String text_pageSize,
            int text_pagesNum , Stringtext_area, String text_page,
String article_title, String article_journalist,
            int clip_filename, java.sql.Timestamp
clip_clippedDatetime, String company_code, String company_name, int
keyword_id,
            String keyword_name, String loginUser_personFullName,
String loginUser_email,String pass, String loginUser_town) throws
SQLException{


        CallableStatement proc_stmt = null;
        try{
          if(connectionObj.Connection()){
            proc_stmt =
connectionObj.getDbConnection().prepareCall("{ call
sp_insertArticle(?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?)
}");
            proc_stmt.setInt(1,pressMedia_id);
            proc_stmt.setString(2,pressMedia_name);
            proc_stmt.setString(3,pressMedia_type);
            proc_stmt.setDate(4,issue_publishedDate);
            proc_stmt.setInt(5,issue_copies);
            proc_stmt.setInt(6,text_id);
            proc_stmt.setString(7,text_ocr);
            proc_stmt.setString(8,text_pageSize);
            proc_stmt.setInt(9,text_pagesNum);
            proc_stmt.setString(10,text_area);
            proc_stmt.setString(11,text_page);
            proc_stmt.setString(12,article_title);
            proc_stmt.setString(13,article_journalist);
            proc_stmt.setInt(14,clip_filename);
            proc_stmt.setTimestamp(15,clip_clippedDatetime);
            proc_stmt.setString(16,company_code);
            proc_stmt.setString(17,company_name);
            proc_stmt.setInt(18,keyword_id);
            proc_stmt.setString(19,keyword_name);
```

```
            proc_stmt.setString(20,loginUser_personFullName);
            proc_stmt.setString(21,loginUser_email);
            proc_stmt.setString(22,pass);
            proc_stmt.setString(23,loginUser_town);

proc_stmt.registerOutParameter(24,java.sql.Types.VARCHAR);

proc_stmt.registerOutParameter(25,java.sql.Types.INTEGER);
            proc_stmt.execute();
        }
        }finally{
            int result = proc_stmt.getInt(25);
            if(!(result==0)){
                String message = "Error inserting article: " +
clip_filename+ ". Error " + proc_stmt.getString(24);
                System.out.println(message);
                proc_stmt.close();
                connectionObj.Close();
                return false;
            }else{
             proc_stmt.close();
             connectionObj.Close();
             return true;
            }
        }
    }

    private boolean insertTender(int pressMedia_id, String
pressMedia_name,String pressMedia_type,
            java.sql.Date issue_publishedDate, int issue_copies, int
text_id , String text_ocr, String text_pageSize,
            int text_pagesNum , Stringtext_area, String text_page,
            int tender_price, String tender_holder, java.sql.Date
tender_expirationDate,
            int clip_filename, java.sql.Timestamp
clip_clippedDatetime, String company_code, String company_name, int
keyword_id,
            String keyword_name, String loginUser_personFullName,
String loginUser_email,String pass, String loginUser_town) throws
SQLException{


        CallableStatement proc_stmt = null;
        try{
            if(connectionObj.Connection()){
                proc_stmt =
connectionObj.getDbConnection().prepareCall("{ call
sp_insertTender(?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?)
}");
                proc_stmt.setInt(1,pressMedia_id);
                proc_stmt.setString(2,pressMedia_name);
                proc_stmt.setString(3,pressMedia_type);
                proc_stmt.setDate(4,issue_publishedDate);
                proc_stmt.setInt(5,issue_copies);
                proc_stmt.setInt(6,text_id);
                proc_stmt.setString(7,text_ocr);
                proc_stmt.setString(8,text_pageSize);
                proc_stmt.setInt(9,text_pagesNum);
                proc_stmt.setString(10,text_area);
                proc_stmt.setString(11,text_page);
                proc_stmt.setInt(12,tender_price);
```

```
                proc_stmt.setString(13,tender_holder);
                proc_stmt.setDate(14,tender_expirationDate);
                proc_stmt.setInt(15,clip_filename);
                proc_stmt.setTimestamp(16,clip_clippedDatetime);
                proc_stmt.setString(17,company_code);
                proc_stmt.setString(18,company_name);
                proc_stmt.setInt(19,keyword_id);
                proc_stmt.setString(20,keyword_name);
                proc_stmt.setString(21,loginUser_personFullName);
                proc_stmt.setString(22,loginUser_email);
                proc_stmt.setString(23,pass);
                proc_stmt.setString(24,loginUser_town);

proc_stmt.registerOutParameter(25,java.sql.Types.VARCHAR);

proc_stmt.registerOutParameter(26,java.sql.Types.INTEGER);
                proc_stmt.execute();
            }
            }finally{
                int result = proc_stmt.getInt(26);
                if(!(result==0)){
                    System.out.println("Error inserting tender: " +
clip_filename+ ". Error " + proc_stmt.getString(25));
                    proc_stmt.close();
                    connectionObj.Close();
                    return false;
                }else{
                 proc_stmt.close();
                 connectionObj.Close();
                 return true;
                }
            }
    }

    private void sendEmail2newClient(String customer_email){
        EmailContainer ec = contHand.getEmailCont();
        EmailFields ef = ec.getEmail(1);
        NewClientEmail nce = new
NewClientEmail(ef,customer_email,initialPassAndUsername);
        Mail m = new Mail(nce);
        m.sendMail();
    }
    private  void sendEmail2admin(ConfigurationContentHandler
contHand,Company company){
        EmailContainer ec = contHand.getEmailCont();
        EmailFields ef = ec.getEmail(3);
        ef.setBody(ef.getBody()+ company.getCode());
        Mail m = new Mail(ef);
            m.sendMail();
    }

}
```

MoveInputFiles.java

```
package pdf2djvu;

import java.io.File;

public class MoveInputFiles extends Thread {
//τα PDF (και XML) αρχεία τα οποία πρέπει να μεταφερθούν.
    Integer files[];
    String redirectionPath;
    String filesInput ;
//    αν η εφαρμογή έχει ρυθμιστεί να κρατάει για επεξεργασία 100
αρχεία
//    (δυάδες αρχείων PDF και XML)
//    όλα τα υπόλοιπα αρχεία (από το 101 και μετά) θα μεταφερθούν.
    int startAt;


    public MoveInputFiles(Integer files[],
ConfigurationContentHandler contHand) {
        super();
        this.files = files;
        this.redirectionPath = contHand.getRedirectionPath();
        filesInput = contHand.getGeneral_input();
        startAt = contHand.getMoveFilesLimit() + 1;
        if(startAt < files.length) this.start();
    }

    @Override
    public void run() {
        File pdfScr;
        File xmlScr;

        File pdfDest ;
        File xmlDest ;


        for (int i = startAt; i<files.length ; i++){
            pdfScr = new File (filesInput + "\\" + files[i] +
".pdf");
            xmlScr = new File (filesInput + "\\" + files[i] +
".xml");
            pdfDest = new File(redirectionPath + "\\" + files[i] +
".pdf");
            xmlDest = new File(redirectionPath + "\\" + files[i] +
".xml");

            xmlScr.renameTo(xmlDest);
            pdfScr.renameTo(pdfDest);
        }//for
    }
}
```

ProducerThread.java

```
package pdf2djvu;

import java.util.concurrent.ArrayBlockingQueue;


public class ProducerThread extends Thread{
    ConfigurationContentHandler contHand;
    ArrayBlockingQueue<Integer> inqueue;
    Integer[] contents = null;

    public ProducerThread(ConfigurationContentHandler contHand,
ArrayBlockingQueue<Integer> inqueue){
        this.contHand = contHand;
        this.inqueue = inqueue;
    }

    public void run() {
        try {
            while(true){
                //Για όσο η Inqueue είναι γεμάτη, περιμένει ένα
δευτερόλεπτο (δλδ. περιμένουμε να αδειάσει)
                while(!inqueue.isEmpty()){
                    Thread.sleep((long)1000);
                }

           //αφού η inqueue πλέον είναι άδεια, παίρνει τα περιεχόμενα
του φακέλου
           while(contents == null){
                //παίρνει τα περιεχόμενα
                contents =
CheckAndGetSortedList.from(contHand.getGeneral_input(),new
PdfExtensionFilter());
                if(contents == null){
                    //αν ο φάκελος δεν έχει αρχεία, περιμένει για
διάστημα ορισμένο στο αρχείο παραμετροποίησης.
                    //(δλδ ελέγχει ανα τακτά χρονικά διαστήματα για
νέα αρχεία)
                    Thread.sleep((long) contHand.getInterval());
                }else{
                    //αν και μόνο αν τα αρχεία προς επεξεργασία στον
φάκελο υπερβαίνουν σε αριθμό
                    //τον επιθυμητό (ο οποίος ορίζεται στο αρχείο
παραμετροποίησης)
                    // μεταφέρει όλα τα επιπλέον αρχεία στην
διεύθυνση που
                    //ορίζει το αρχείο παραμετροποίησης για να
επεξεργαστούν από
                    //άλλο "κλώνο" της εφαρμογής.
                    new MoveInputFiles(contents, contHand);
                }
           }

           //αφού βρέθηκαν αρχεία προς επεξεργασία, εισάγει το όνομα
των αρχείων στο inqueue
           insertIntoQueue();
```

```
            }//while
          } catch (InterruptedException ex) {
            System.out.println(ex.getMessage());
          }
      }//run()

      private void insertIntoQueue() throws InterruptedException{
          for (int i = 0; i < inqueue.size(); i++) {
              inqueue.put(contents[i]);
          }
          contents = null;
      }
}
```

ProductionFiles.java

```
package pdf2djvu;

import java.io.File;
import java.util.Calendar;
import java.util.Hashtable;


public class ProductionFiles {
    private String  PdfExt = ".pdf";
    private String  DjvuExt = ".djvu";
    private String  XmlExt = ".xml";

    private String[] generalInputFiles = new String[3];
    private String[] onErrorFiles = new String[3];
    private String[] destinationFiles = new String[3];

    private ConfigurationContentHandler contHand ;


    private String filenameNumber ;

    Hashtable fileExistance;


    public ProductionFiles (ConfigurationContentHandler contHand, int
filenameNumber){
        this.contHand = contHand;

        this.filenameNumber = String.valueOf(filenameNumber);

        setGeneralInputFiles();
        setOnErrorFiles();
        setDestFiles();
    }

    private void setGeneralInputFiles(){

        String input = contHand.getGeneral_input();
        //xml
        File tmp = new File(input + "\\" + filenameNumber + XmlExt);
        File tmp2UpExt = new File(input + "\\" + filenameNumber +
XmlExt.toUpperCase());
        tmp2UpExt.renameTo(tmp);
        generalInputFiles[0] = input + "\\" + filenameNumber +
XmlExt;
        //pdf
        tmp = new File(input + "\\" + filenameNumber + PdfExt);
        tmp2UpExt = new File(input + "\\" + filenameNumber +
PdfExt.toUpperCase());
        tmp2UpExt.renameTo(tmp);
        generalInputFiles[1] = input + "\\" + filenameNumber +
PdfExt;
        //djvu
        tmp = new File(input + "\\" + filenameNumber + DjvuExt);
        tmp2UpExt = new File(input + "\\" + filenameNumber +
DjvuExt.toUpperCase());
```

```
        tmp2UpExt.renameTo(tmp);
        generalInputFiles[2] = input + "\\" + filenameNumber +
DjvuExt;

    }



    private void setOnErrorFiles(){
        String onerrorPath = contHand.getOutput_onError();
        //xml
        onErrorFiles[0] = onerrorPath + "\\" + filenameNumber +
XmlExt;
        //pdf
        onErrorFiles[1] = onerrorPath + "\\" + filenameNumber +
PdfExt;
        //djvu
        onErrorFiles[2] = onerrorPath + "\\" + filenameNumber +
DjvuExt;
    }


    private void setDestFiles(){

        int FileNameInInteger = Integer.parseInt(filenameNumber);
        String TargetFolder=String.valueOf(FileNameInInteger/200);

        //xml
        String xmlOutput = contHand.getXml_output();

        Calendar currDate = Calendar.getInstance();
        @SuppressWarnings("static-access")
        String folderName =
String.valueOf(currDate.get(currDate.DAY_OF_MONTH))+"_"+
                String.valueOf(currDate.get(currDate.MONTH)+1) + "_"+
                String.valueOf(currDate.get(currDate.YEAR));
        File foldPath = new File(xmlOutput + "\\" + folderName);
        if(!foldPath.exists()) foldPath.mkdir();

        destinationFiles[0] = xmlOutput + "\\" + folderName + "\\" +
filenameNumber + XmlExt;

        //pdf
        String pdfOutput = contHand.getPdf_output();
        File PdfFolderPath=new File(pdfOutput+"\\"+TargetFolder);
        if(!PdfFolderPath.exists()) PdfFolderPath.mkdir();

        destinationFiles[1] = pdfOutput + "\\" + TargetFolder + "\\"
+  filenameNumber + PdfExt;

        //djvu
        String djvuOutput = contHand.getDjvu_output();
        File DjvuFolderPath = new File(djvuOutput+"\\"+TargetFolder);
        if(!DjvuFolderPath.exists()) DjvuFolderPath.mkdir();

        destinationFiles[2] = djvuOutput + "\\" + TargetFolder + "\\"
+  filenameNumber + DjvuExt;
    }


    private void checkFilesLocation(){
```

```
        fileExistance = new Hashtable();
        //xml

fileExistance.put("input_xml",(boolean)checkFileLocation(generalInput
Files[0]));
        //pdf

fileExistance.put("input_pdf",(boolean)checkFileLocation(generalInput
Files[1]));
        //djvu

fileExistance.put("input_djvu",(boolean)checkFileLocation(generalInpu
tFiles[2]));
         //xml

fileExistance.put("dest_xml",(boolean)checkFileLocation(destinationFi
les[0]));
        //pdf

fileExistance.put("dest_pdf",(boolean)checkFileLocation(destinationFi
les[1]));
        //djvu

fileExistance.put("dest_djvu",(boolean)checkFileLocation(destinationF
iles[2]));

    }
    private boolean checkFileLocation(String unc){
        File tmp = new File(unc);
        return (tmp.exists())?( true) : ( false);
    }

    public int getActionOption(){
        checkFilesLocation();
        if((Boolean)fileExistance.get("input_xml") &&
(Boolean)fileExistance.get("input_pdf")
                && (Boolean)fileExistance.get("input_djvu")) {
            return 3;
        }
        else if((Boolean)fileExistance.get("input_xml") &&
(Boolean)fileExistance.get("input_pdf")){
            return 2;
        }
        else return 0;
    }


     public ConfigurationContentHandler getConfigurContentHand() {
        return contHand;
    }

    public String getXmlDestFile(){
        return destinationFiles[0];
    }
    public String getPdfDestFile(){
        return destinationFiles[1];
    }
    public String getDjvuDestFile(){
        return destinationFiles[2];
    }
```

```
    public String getXmlGeneralInputFile(){
        return generalInputFiles[0];
    }
    public String getPdfGeneralInputFile(){
        return generalInputFiles[1];
    }
    public String getDjvuGeneralInputFile(){
        return generalInputFiles[2];
    }

    public String getXmlOnErrorFile(){
        return onErrorFiles[0];
    }
     public String getPdfOnErrorFile(){
        return onErrorFiles[1];
    }


    public void moveFiles(int option, int ConvertResult, boolean
successfulUpdate){
       if(option == 2){
            if(ConvertResult == 0){
                if(successfulUpdate){
                    //xml, pdf στην τελική τους διεύθυνση.
                    //Το Djvu δημιουργείτε απευθείας στην τελική
διεύθυνση.
                    File xmlIn = new File(generalInputFiles[0]);
                    File xmlOut = new File(destinationFiles[0]);
                    while(xmlIn.exists() && !xmlOut.exists()){
                        xmlIn.renameTo(xmlOut);
                    }

                    File pdfIn = new File(generalInputFiles[1]);
                    File pdfOut = new File(destinationFiles[1]);
                    while(pdfIn.exists() && !pdfOut.exists()){
                        pdfIn.renameTo(pdfOut);
                    }
                }
                else{
                    //xml,pdf,djvu to Error folder
                    File xmlIn = new File(generalInputFiles[0]);
                    File xmlOut = new File(onErrorFiles[0]);
                    while(xmlIn.exists() && !xmlOut.exists()){
                        xmlIn.renameTo(xmlOut);
                    }

                    File pdfIn = new File(generalInputFiles[1]);
                    File pdfOut = new File(onErrorFiles[1]);
                    while(pdfIn.exists() && !pdfOut.exists()){
                        pdfIn.renameTo(pdfOut);
                    }

                    File djvuIn = new File(destinationFiles[2]);
                    File djvuOut = new File(onErrorFiles[2]);
                    while(djvuIn.exists()&& !djvuOut.exists()){
                        djvuIn.renameTo(djvuOut);
                    }
                }
            }
            else{
                //xml,pdf to error folder
```

```
                File xmlIn = new File(generalInputFiles[0]);
                File xmlOut = new File(onErrorFiles[0]);
                while(xmlIn.exists() && !xmlOut.exists()){
                    xmlIn.renameTo(xmlOut);
                }

                File pdfIn = new File(generalInputFiles[1]);
                File pdfOut = new File(onErrorFiles[1]);
                while(pdfIn.exists() && !pdfOut.exists()){
                    pdfIn.renameTo(pdfOut);
                }
            }
        }
        else if(option == 3){
            if(successfulUpdate){
               //xml,pdf and djvu to dest folder
                File xmlIn = new File(generalInputFiles[0]);
                File pdfIn = new File(generalInputFiles[1]);
                File djvuIn = new File(generalInputFiles[2]);

                File xmlOut = new File(destinationFiles[0]);
                File pdfOut = new File(destinationFiles[1]);
                File djvuOut = new File(destinationFiles[2]);

                while(xmlIn.exists() && !xmlOut.exists()){
                 xmlIn.renameTo(xmlOut);
                }
                while(pdfIn.exists()&& !pdfOut.exists()){
                 pdfIn.renameTo(pdfOut);
                }
                while(djvuIn.exists()&& !djvuOut.exists()){
                 djvuIn.renameTo(djvuOut);
                }

            }
            else{
              //xml,pdf and djvu to error folder
                File xmlIn = new File(generalInputFiles[0]);
                File pdfIn = new File(generalInputFiles[1]);
                File djvuIn = new File(generalInputFiles[2]);

                File xmlOut = new File(onErrorFiles[0]);
                File pdfOut = new File(onErrorFiles[1]);
                File djvuOut = new File(onErrorFiles[2]);

               while(xmlIn.exists() && !xmlOut.exists()){
                xmlIn.renameTo(xmlOut);
               }
               while(pdfIn.exists()&& !pdfOut.exists()){
                pdfIn.renameTo(pdfOut);
               }
               while(djvuIn.exists()&& !djvuOut.exists()){
                djvuIn.renameTo(djvuOut);
               }
            }
        }
        else{
        }
    }
}
```

Convert.java

```
package pdf2djvu;

import java.io.File;
import java.lang.ProcessBuilder.Redirect;


public class Convert{

private ProductionFiles productionFiles;
private ConfigurationContentHandler contHand;
private int exitValue;

    public Convert(ProductionFiles productionFiles){
        this.productionFiles = productionFiles;
        contHand = productionFiles.getConfigurContentHand();
        run();
    }

    private void run() {
        // Στην περίπτωση που υπάρχει το djvu αρχείο, γίνεται
overight. Ο παρακάτω έλεγχος
        //γίνεται για να το αποφύγουμε. Αυτό θα έχει ως αποτέλεσμα να
μην γίνει το update στη βάση
        //και να μετακινηθούν τα αρχεία στο onError.
        if(!new File(productionFiles.getDjvuDestFile()).isFile()){
            exitValue =
convertToDjVu(productionFiles.getPdfGeneralInputFile(),productionFile
s.getDjvuDestFile(),(contHand.getLogFiles())[0][0]);
        }
        else{
            exitValue = -1;
        }
        if(exitValue != 0){//κάτι πήγε στραβά κατά την μετατροπή.
            EmailContainer ec = contHand.getEmailCont();
            EmailFields ef = new EmailFields(ec.getEmail(0));
            ef.setBody(ef.getBody()+
productionFiles.getPdfGeneralInputFile());
            Mail m = new Mail(ef);
            m.sendMail();
        }
        else{//όλα πήγαν καλά κατά την μετατροπή.
        }
    }


  private  int convertToDjVu(String pdf, String djvu,String log){
//        String profile = "--profile=convert";
//        String command = "documenttodjvu.exe";
        String command = contHand.getconvertCommand1();
        String profile = contHand.getConvertProfile();
        String[] CommandArray = {command,profile, pdf, djvu};
         Process pr;
         ProcessBuilder pb;
         try {
                pb = new ProcessBuilder(CommandArray);
                File f = new File(log);
```

```
                f.createNewFile();

                pb.redirectOutput(Redirect.to(f));
                pb.redirectErrorStream(true);

                pr = pb.start();
                pr.waitFor();
                System.out.println("Attempt 1 : "+pdf+" has exit
value " + pr.exitValue());

                //Αν η μετατροπή δεν έγινε επιτυχώς, διέγραψε το
corrupted File.
                if(pr.exitValue()!=0){
                    File corrunptedFile = new File(djvu);
                    while(corrunptedFile.exists()){
                        if(corrunptedFile.length()<2){
                            corrunptedFile.delete();
                        }
                    }
                }

                //Αν η μετατροπή δεν ολοκληρώθηκε επιτυχώς,
προσπάθησε με το δεύτερο πρόγραμμα.
                if(pr.exitValue()!=0){
                    Process pr2;
                    ProcessBuilder pb2;

                    //Η πλήρης διεύθυνση του .EXE είναι απαραίτητο -
αλλιώς δεν τρέχει.
                    String[] Command2 = new
String[contHand.getOptions().length+4];
                    Command2[0] = contHand.getConvertCommand2();
                    Command2[1] = "-o";
                    Command2[2] = djvu;
                    Command2[Command2.length-1] = pdf;
                    int j=0;
                    for(int i=3 ; i<Command2.length-1; i++){
                        Command2[i] = contHand.getOptions()[j];
                        j++;
                    }

                    pb2 = new ProcessBuilder(Command2);

                    File f2 = new File(log.substring(0,log.length() -
4) + "2.txt");
                    f2.createNewFile();

                    pb2.redirectOutput(Redirect.to(f2));
                    pb2.redirectErrorStream(true);

                    pr2 = pb2.start();
                    pr2.waitFor();

                    System.out.println("Attempt 2 : "+pdf+" has exit
value " + pr2.exitValue());

                    return pr2.exitValue();

                }
                return pr.exitValue();
```

```
                }catch(Exception e) {
                    System.out.println(e.toString());
                    e.printStackTrace();
                    return -10;
            }
    }//convertToDjVu

     public int getExitValue() {
         return exitValue;
     }

}//Convert
```

UpdateDatabase .java

```
package pdf2djvu;

import java.io.FileNotFoundException;
import java.io.IOException;
import java.io.UnsupportedEncodingException;
import java.sql.SQLException;

public class UpdateDatabase  {


    private String xmlFile;
    private ConfigurationContentHandler contHand = null;
    private DistributionXmlObj xmlObj = null;
    private boolean updateResult = false;

    public UpdateDatabase(ProductionFiles ifile) {

        this.xmlFile =ifile.getXmlGeneralInputFile();
        this.contHand = ifile.getConfigurContentHand();

        if(xmlFile != null) this.start();

    }

    public boolean getUpdateResult(){
        return updateResult;
    }

    public void start() {
       ReadDistributionXML rdXML = null;
              try {
                    rdXML = new ReadDistributionXML(xmlFile);
                    xmlObj = rdXML.getDistributionXmlObj();
                    SQLDatabaseQueries sql_db = new
SQLDatabaseQueries(contHand);
                    if(sql_db.insertClip(xmlObj)) {
                        updateResult= true;
                    }else{
                        updateResult= false;
                    }

                } catch (FileNotFoundException ex) {
                    System.out.println("FileNotFoundException while
trying to read Distribution XML file: " + xmlFile + ". Error:" +
ex.getMessage());
                } catch (UnsupportedEncodingException ex) {
                    System.out.println("UnsupportedEncodingException
while trying to read Distribution XML file: " + xmlFile + ". Error:"
+ ex.getMessage());
                }catch (SQLException ex){
                    System.out.println("SQLException inserting Clip:
" + xmlObj.getId() + ". Error: " + ex.getMessage().toString());
                    updateResult = false;
                    //send mail
                    sendEmail2admin();
                }
```

```
                catch (IOException ex) {
                    System.out.println("IOException while trying to
read Distribution XML file: " + xmlFile + ". Error:" +
ex.getMessage());
                    System.out.println(ex.getMessage());
                }

    }

    private void sendEmail2admin(){
        EmailContainer ec = contHand.getEmailCont();
        EmailFields ef = ec.getEmail(2);
        ef.setBody(ef.getBody()+ "\n\r" + xmlFile);
        Mail m = new Mail(ef);
            m.sendMail();
    }
}
```

ConsumerThread.java

```
package pdf2djvu;

import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;
import java.util.concurrent.ArrayBlockingQueue;

public class ConsumerThread extends Thread{

    ConfigurationContentHandler contHand ;
    Integer filenameNumber;
    ArrayBlockingQueue<Integer> inqueue;

    @SuppressWarnings("static-access")
    public ConsumerThread(ConfigurationContentHandler contHand,
ArrayBlockingQueue inqueue){
       this.contHand = contHand;
       this.inqueue = inqueue;
    }

    public  void run(){
        filenameNumber = -1;
        int i = 0;
        int option = -1 ;
        boolean updateResult=false;
        int convertResult = -1;
        ProductionFiles productionFilesPaths = null;

      try{
        while(true){
            filenameNumber = -1;
            i = 0;
            option = -1 ;
            updateResult=false;
            convertResult = -1;
            productionFilesPaths = null;

               filenameNumber = (Integer) inqueue.take();

               System.out.println(this.getName()+"----  input file
number: " + filenameNumber);
               productionFilesPaths = new ProductionFiles(contHand,
filenameNumber);

               option = productionFilesPaths.getActionOption();
               System.out.println(this.getName()+"Input option : " +
option + " (2=>xml,pdf | 3=>xml,pdf,djvu)");

               if(option == 2){
                   Convert con = new Convert(productionFilesPaths);
                   convertResult = con.getExitValue();
                   //εισαγωγή δεδομένων στη βάση
                   if(convertResult == 0){
                       UpdateDatabase ud = new
UpdateDatabase(productionFilesPaths);
```

```
                        updateResult = ud.getUpdateResult();
                        System.out.println(this.getName()+" Database
Update: " + updateResult);
                    }
                    else{
                        //write the output of the convert process in
the log file
                        logTheUnsuccessfullConverOutput(contHand);
                    }
                }
                else if (option == 3){
                    //εισαγωγή δεδομένων στη βάση
                    UpdateDatabase ud = new
UpdateDatabase(productionFilesPaths);
                    updateResult = ud.getUpdateResult();
                    System.out.println(this.getName()+" Database
Update: " + updateResult);
                }
                else{
                    System.out.println("Invalid option.");
                }
            i++;

        }
     }catch (InterruptedException ex) {
         String message = this.getName()+" Interrupted Exception
caught. Error: " + ex.getMessage();
         System.out.println(message);
     }catch(Exception exc){
          String message = "Other exception (inserting XML):" +
exc.getMessage();
          System.out.println(message);
          sendEmail2admin(contHand, message);
      }finally{
          productionFilesPaths.moveFiles(option, convertResult,
updateResult);
      }
    }


    private void
logTheUnsuccessfullConverOutput(ConfigurationContentHandler
contHand){
        FileReader fr =  null;
        try {
            File outputFile = new File(contHand.getLogFiles()[0][0]);
            fr = new FileReader(outputFile);
            int charac = fr.read();
            String output = String.valueOf((char)charac);
            System.out.println(this.getName()+"Convertion output
BEGIN:___________");
            while(charac != -1){
                output += String.valueOf((char)charac);
                charac = fr.read();
            }
             System.out.println(this.getName()+output);
        } catch (FileNotFoundException ex) {
             System.out.println(this.getName()+ex.getMessage());
        }catch (IOException ex) {
             System.out.println(this.getName()+ex.getMessage());
        } finally {
```

```
            try {
                System.out.println(this.getName()+"Convertion output
END:_____________");
                fr.close();
            } catch (IOException ex) {
               System.out.println(this.getName()+ex.getMessage());
            }
        }
    }

      private  void sendEmail2admin(ConfigurationContentHandler
contHand, String message){
        EmailContainer ec = contHand.getEmailCont();
        EmailFields ef = ec.getEmail(4);
        ef.setBody(message);
        Mail m = new Mail(ef);
            m.sendMail();
    }

}
```

AccountsSynchr.java

```
package pdf2djvu;

import java.io.BufferedWriter;
import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.OutputStreamWriter;
import java.sql.SQLException;
import java.util.Calendar;
import java.util.Date;
import java.util.TimerTask;
import java.util.Vector;

public class AccountsSynchr extends TimerTask{

    ConfigurationContentHandler contHand;
    SQLDatabaseQueries sql_db;
    OracleDatabaseQueries oracle_db;

    public AccountsSynchr(ConfigurationContentHandler contHand) {
        this.contHand = contHand;
        sql_db = new SQLDatabaseQueries(contHand);
        oracle_db = new OracleDatabaseQueries(contHand);
    }

    @SuppressWarnings("static-access")
    public void run(){
        writeToSynchronizationLogFile("Synchronization started.");

//        Έλεγχος για το αν είναι enabled η διαδικασία του
συγχρονισμού στο configuration File
        String enabled = contHand.getOracle_mssql_synch_enabled();
        if(enabled.equals("false")) {
            writeToSynchronizationLogFile("Synchronization is
Disabled.");
        }else{

            //A. Εύρεση και δημιουργία νέων πελατών
            //A.0 - προδιεργασία - Πάρε όλους τους πελάτες που
υπάρχουν στην βάση του υποσυστήματος δημοσίευσης.
             try{
                String webCustomersStr = sql_db.getAllCustomersStr();
             //A.1 Βρες τους νέους πελάτες (= οι πελάτες που υπάρχουν
στη βάση του παραγωγικού υποσυστήματος και δεν υπάρχουν στην βάση του
υποσυστήματος δημοσίευσης).
                if(webCustomersStr == null){
                    webCustomersStr = "('nothing')";
                }
                 Vector<Company> newCustomers =
oracle_db.getNewCustomers(webCustomersStr);

                 //A.2 Δημιουργία των νέων πελατών στο υποσύστημα
δημοσίευσης
                 if(newCustomers!=null){
                      writeToSynchronizationLogFile("New accounts
creation:");
```

```
                    for(int i=0 ; i<newCustomers.size(); i++){
                        if(newCustomers.get(i) == null ){continue;}

if(sql_db.createAccount(newCustomers.get(i),true)){

writeToSynchronizationLogFile(newCustomers.get(i).getCode());
                      }else{

writeToSynchronizationLogFile(newCustomers.get(i).getCode() + "
(error(s) on creation! Check log file for details.");
                      }

                    }
                }


            //B. Ενεργοποίηση πελατών που απενεργοποιήθηκαν για
δεδομένο χρονικό διάστημα.
            //B.0 - προδιεργασία - Πάρε τους πελάτες που δεν είναι
ενεργοί πλέον από την βάση του υποσυστήματος δημοσίευσης.
             String  webStoppedCustomersStr =
sql_db.getStoppedCustomersStr();

            //B.1 Βρές τους πελάτες που πρέπει να ενεργοποιηθούν
στην βάση του υποσυστήματος δημοσίευσης.
             if( webStoppedCustomersStr ==null ){
                 webStoppedCustomersStr = "('nothing')";
             }
                 String[] custToActivate =
oracle_db.getCustomersToActivate(webStoppedCustomersStr);

                 //B.2 ενεργοποίηση πελατών στο υποσύστημα
δημοσίευσης.
                 if(custToActivate!=null){
                     writeToSynchronizationLogFile("Accounts
activation:");
                     for(int i = 0 ; i < custToActivate.length;
i++){

if(sql_db.activateCompany(custToActivate[i])){

writeToSynchronizationLogFile(custToActivate[i]);
                        }else{

writeToSynchronizationLogFile(custToActivate[i] + " (error on account
acitvation! Check log file for details.");
                        }

                     }
                 }


            //C Απενεργοποίηση των πελατών που σταματούν τη
συνδρομή.
            //C.0 - προδιεργασία - Πάρε τους ενεργούς πελάτες που
πρέπει να απενεργοποιηθούν στην βάση του υποσυστήματος δημοσίευσης.
             String activeOnlyClipCustomers =
sql_db.getActiveCustomersStr();

            //C.1 Πάρε τους πελάτες που πρέπει να απενεργοποιηθούν
στην βάση του υποσυστήματος δημοσίευσης.
```

```
                if(activeOnlyClipCustomers != null){
                    String[] custToDeact =
oracle_db.getCustomersToDeactiv(activeOnlyClipCustomers);

                    //C.2 Απενεργοποίηση πελατών στη βάση του
υποσυστήματος δημοσίευσης
                    if(custToDeact!=null){
                        writeToSynchronizationLogFile("Deactivate
accounts:");
                        for(int i = 0 ; i < custToDeact.length; i++){
                            sql_db.deactivateCompany(custToDeact[i]);

if(sql_db.activateCompany(custToActivate[i])){

writeToSynchronizationLogFile(custToActivate[i]);
                            }else{

writeToSynchronizationLogFile(custToActivate[i] + " (error on account
deacitvation! Check log file for details.");
                            }
                        }
                    }
                }

              //D.1 Πάρε τους ενεργούς πελάτες από την βάση του
παραγωγικού υποσυστήματος

                    webCustomersStr = sql_db.getActiveCustomersStr();
                    if(webCustomersStr == null){
                        webCustomersStr = "('nothing')";
                    }
                    Vector<Company> activeCustomers =
oracle_db.getActiveCustomersKeywords(webCustomersStr);

                     //D. Συγχρονισμός των λέξεων κλειδιών
                     if(activeCustomers!=null){
                          writeToSynchronizationLogFile("Reset
Keywords:");
                         for(int i=0 ; i<activeCustomers.size(); i++){
                            if(activeCustomers.get(i) == null
){continue;}

if(sql_db.createAccount(activeCustomers.get(i),false)){

writeToSynchronizationLogFile(activeCustomers.get(i).getCode());
                            }else{

writeToSynchronizationLogFile(activeCustomers.get(i).getCode() + "
(error(s) on kewords synchronization! Check log file for details.");
                            }
                        }
                    }

                 writeToSynchronizationLogFile("Synchronization Done.");

              }catch(SQLException exc){
                    String message = "SQLException in Accounts
Synchronization Thread: " + exc.getMessage();
                     writeToSynchronizationLogFile(message);
                     sendEmail2admin(contHand, message);
               }catch(Exception exc){
```

```
                String message = "Exception in Accounts
Synchronization Thread: " + exc.getMessage();
                writeToSynchronizationLogFile(message);
                sendEmail2admin(contHand, message);
            }

    }
    }

    public Date getTime(){
        String time = contHand.getOracle_mssql_synch_start_time();
        Calendar calendar = Calendar.getInstance();
        calendar.set(Calendar.HOUR_OF_DAY, Integer.valueOf(time));
        calendar.set(Calendar.MINUTE, 0);
        return calendar.getTime();
    }

    public long getInterval(){
        return
Integer.valueOf(contHand.getOracle_mssql_synch_interval()) * 60 * 60
* 1000;
    }
    public void writeToSynchronizationLogFile(String message){
      try {
          Calendar calendar = Calendar.getInstance();
          int day = calendar.get(Calendar.DAY_OF_MONTH);
          int month = calendar.get(Calendar.MONTH) + 1;
          int year = calendar.get(Calendar.YEAR);
          int hour = calendar.get(Calendar.HOUR_OF_DAY);
          int min = calendar.get(Calendar.MINUTE);

          String timestamp = String.valueOf(day)+"/"+
String.valueOf(month)+"/"+ String.valueOf(year)+" "+
String.valueOf(hour)+":"+ String.valueOf(min);

        if(contHand.getLogFiles()[2]!=null){
            File synchLogFile = new
File(contHand.getLogFiles()[2][0]);
            if(!synchLogFile.isFile()){
                synchLogFile.createNewFile();
            }
            BufferedWriter out = new BufferedWriter(new
OutputStreamWriter(new
FileOutputStream(synchLogFile,Boolean.valueOf(contHand.getLogFiles()[
2][3])),"UTF-8"));

            out.write(timestamp + " " +  message);
            out.newLine();
            out.close();
        }

        } catch (IOException ex) {
            System.out.println(ex.getMessage());
        } catch(IndexOutOfBoundsException e){
            System.out.println(e.getMessage());
        }

    }

    private  void sendEmail2admin(ConfigurationContentHandler
contHand, String message){
```

```
        EmailContainer ec = contHand.getEmailCont();
        EmailFields ef = ec.getEmail(4);
        ef.setBody(message);
        Mail m = new Mail(ef);
            m.sendMail();
    }
}
```

Start.java

```
package pdf2djvu;

import java.util.Date;
import java.util.Timer;
import java.util.concurrent.ArrayBlockingQueue;


public class Start {
     ConfigurationContentHandler contHand;
    int[] contents = null;
    int numberOfThreads;
    ArrayBlockingQueue<Integer> inqueue;
    int constant = 4 ; // to megethos tis ouras einai pollaplasio
aytoy toy megethous
    ProducerThread prodThread;
    ConsumerThread[] threads;
    Timer timer;

    public Start(String UserXmlConfFilePath) {
    if(!UserXmlConfFilePath.equals("")){
        ReadXMLConf input = new ReadXMLConf(UserXmlConfFilePath);
        contHand = input.getMyContentHandler();
        boolean start = true;
        try {

LoggingInitialization.Initialize(contHand.getLogFiles()[1][0],Integer
.parseInt(contHand.getLogFiles()[1][1]),Integer.parseInt(contHand.get
LogFiles()[1][2]),Boolean.valueOf(contHand.getLogFiles()[1][3]));
        } catch (Exception ex) {
            start = false;
        }
        if(start){
            numberOfThreads = contHand.getNumberOfThreads();
            inqueue = new
ArrayBlockingQueue<Integer>(numberOfThreads*constant);
             //start the fill in thread
            prodThread = new ProducerThread(contHand, inqueue);
            prodThread.start();


//             start the process threads
            threads = new ConsumerThread[numberOfThreads];
            for(int i=0; i<threads.length; i++){
                threads[i] = new ConsumerThread(contHand, inqueue);
                threads[i].start();
            }

            timer = new Timer();
            AccountsSynchr accounts_synchronization = new
AccountsSynchr(contHand);
            Date start_time = accounts_synchronization.getTime();
            long interval = accounts_synchronization.getInterval();
            timer.scheduleAtFixedRate(accounts_synchronization,
start_time, interval);
        }
        else{
```

```
                System.exit(1);
            }
        }

    }

    public void stopExecution(){
        prodThread.interrupt();
        for(int i=0 ; i< threads.length; i++){
            threads[i].interrupt();
        }
        timer.cancel();
    }
}
```

ConvertApplicUI.java

```
package pdf2djvu;

import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;
import javax.swing.JFileChooser;
import javax.swing.JOptionPane;

public class ConvertApplicUI extends javax.swing.JFrame {

   Start st;

    /** Creates new form ConvertApplicUI */
    public ConvertApplicUI() {
        initComponents();
    }

    private void initComponents() {

        jTabbedPane1 = new javax.swing.JTabbedPane();
        ConfigPane = new javax.swing.JPanel();
        jPanel4 = new javax.swing.JPanel();
        StartBut = new javax.swing.JButton();
        StopBut = new javax.swing.JButton();
        Exit2 = new javax.swing.JButton();
        jPanel2 = new javax.swing.JPanel();
        jLabel1 = new javax.swing.JLabel();
        jLabel2 = new javax.swing.JLabel();
        PathTxtField = new javax.swing.JTextField();
        setConfigFilePath();
        Browse = new javax.swing.JButton();


setDefaultCloseOperation(javax.swing.WindowConstants.DO_NOTHING_ON_CL
OSE);
        setTitle("Convert, Insert, Synch Application V5.4 (PDF
version)");

        jTabbedPane1.setFont(new java.awt.Font("Times New Roman", 1,
14));

        StartBut.setLabel("<html>\n<font
size=4><b>Εναρξη</b></font>\n</html>");
        StartBut.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                StartButActionPerformed(evt);
            }
        });

        StopBut.setEnabled(false);
        StopBut.setLabel("<html>\n<font
size=4><b>Τερματισμός</b></font>\n</html>");
```

```
        StopBut.addActionListener(new java.awt.event.ActionListener()
{
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                StopButActionPerformed(evt);
            }
        });

        Exit2.setText("<html>\n<font
size=4><b>Έξοδος</b></font>\n</html>");
        Exit2.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                Exit2ActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout jPanel4Layout = new
javax.swing.GroupLayout(jPanel4);
        jPanel4.setLayout(jPanel4Layout);
        jPanel4Layout.setHorizontalGroup(

jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.L
EADING)
            .addGroup(jPanel4Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(StartBut,
javax.swing.GroupLayout.PREFERRED_SIZE, 88,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(37, 37, 37)
                .addComponent(StopBut,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
315, Short.MAX_VALUE)
                .addComponent(Exit2,
javax.swing.GroupLayout.PREFERRED_SIZE, 98,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap())
        );
        jPanel4Layout.setVerticalGroup(

jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.L
EADING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel4Layout.createSequentialGroup()

.addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)

.addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.A
lignment.BASELINE)
                    .addComponent(StartBut,
javax.swing.GroupLayout.PREFERRED_SIZE, 39,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(StopBut,
javax.swing.GroupLayout.PREFERRED_SIZE, 39,
javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
                    .addComponent(Exit2,
javax.swing.GroupLayout.PREFERRED_SIZE, 40,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addContainerGap())
        );


jPanel2.setBorder(javax.swing.BorderFactory.createEtchedBorder());

        jLabel1.setText("Διαλέξτε αρχείο ρυθμίσεων:");

        jLabel2.setText("Διεύθυνση:");

        PathTxtField.setToolTipText("");

        Browse.setText("Αναζήτηση...");
        Browse.addActionListener(new java.awt.event.ActionListener()
{
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                BrowseActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout jPanel2Layout = new
javax.swing.GroupLayout(jPanel2);
        jPanel2.setLayout(jPanel2Layout);
        jPanel2Layout.setHorizontalGroup(

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.L
EADING)
            .addGroup(jPanel2Layout.createSequentialGroup()

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.A
lignment.LEADING)
                    .addGroup(jPanel2Layout.createSequentialGroup()
                        .addGap(22, 22, 22)
                        .addComponent(jLabel2)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(PathTxtField,
javax.swing.GroupLayout.PREFERRED_SIZE, 491,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(18, 18, 18)
                        .addComponent(Browse))
                    .addGroup(jPanel2Layout.createSequentialGroup()
                        .addContainerGap()
                        .addComponent(jLabel1)))
                .addContainerGap(25, Short.MAX_VALUE))
        );
        jPanel2Layout.setVerticalGroup(

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.L
EADING)
            .addGroup(jPanel2Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jLabel1)
                .addGap(21, 21, 21)

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.A
lignment.BASELINE)
```

```
                    .addComponent(jLabel2)
                    .addComponent(PathTxtField,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(Browse))
                .addGap(23, 23, 23))
        );

        javax.swing.GroupLayout ConfigPaneLayout = new
javax.swing.GroupLayout(ConfigPane);
        ConfigPane.setLayout(ConfigPaneLayout);
        ConfigPaneLayout.setHorizontalGroup(

ConfigPaneLayout.createParallelGroup(javax.swing.GroupLayout.Alignmen
t.LEADING)
            .addGroup(ConfigPaneLayout.createSequentialGroup()
                .addGap(22, 22, 22)

.addGroup(ConfigPaneLayout.createParallelGroup(javax.swing.GroupLayou
t.Alignment.LEADING)
                    .addComponent(jPanel4,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jPanel2,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addContainerGap(20, Short.MAX_VALUE))
        );
        ConfigPaneLayout.setVerticalGroup(

ConfigPaneLayout.createParallelGroup(javax.swing.GroupLayout.Alignmen
t.LEADING)
            .addGroup(ConfigPaneLayout.createSequentialGroup()
                .addGap(19, 19, 19)
                .addComponent(jPanel2,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED
)
                .addComponent(jPanel4,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(66, 66, 66))
        );

        jTabbedPane1.addTab("Ρυθμίσεις...", ConfigPane);

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
```

```
                .addComponent(jTabbedPane1,
javax.swing.GroupLayout.PREFERRED_SIZE, 762,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jTabbedPane1,
javax.swing.GroupLayout.PREFERRED_SIZE, 239,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );

        pack();
    }// </editor-fold>

    @SuppressWarnings("static-access")
    private void StartButActionPerformed(java.awt.event.ActionEvent
evt) {
        if(!PathTxtField.getText().equals("")){
            String configurationFilePath = PathTxtField.getText();
            st = new Start(configurationFilePath);
            StartBut.setEnabled(false);
            StopBut.setEnabled(true);
            Exit2.setEnabled(false);
        }
        else{
            JOptionPane.showMessageDialog(jTabbedPane1, "Παρακαλώ
επιλέξτε αρχείο ρυθμίσεων.");
        }
}

    @SuppressWarnings("static-access")
    private void StopButActionPerformed(java.awt.event.ActionEvent
evt) {
        StartBut.setEnabled(true);
        StopBut.setEnabled(false);
        Exit2.setEnabled(true);
//      αν το st thread είναι σε sleep mode πρέπει να γίνει stop.
        st.stopExecution();

}

    private void Exit2ActionPerformed(java.awt.event.ActionEvent evt)
{
        System.exit(0);
}

    private void BrowseActionPerformed(java.awt.event.ActionEvent
evt) {
    final JFileChooser fc4Open = new JFileChooser();
    int returnVal = fc4Open.showOpenDialog(jTabbedPane1);
    if(returnVal == JFileChooser.APPROVE_OPTION){
        File selectedFile = fc4Open.getSelectedFile();
```

```
        PathTxtField.setText(selectedFile.getPath());
    }
}

    /**
     * @param args the command line arguments
     */

    //αυτή η μέθοδος χρησημοποιείται στο 'post-creation code' του
PathTxtField.
    //Διαβάζει την διεύθυνση που υπάρχει καταγεγραμμένη στο αρχείο
ConfigurationLocation.txt.
    private void setConfigFilePath(){
         FileReader fr = null;
         File f = null;
            try {
                f = new File("C:\\....\\ConfigurationLocation.txt");

                fr = new FileReader(f);
          } catch (FileNotFoundException ex) {
                JOptionPane.showMessageDialog(jTabbedPane1,"Το αρχείο
παραμετροποίησης δεν βρέθηκε. Ορίστε άλλη διεύθυνση \n\rστο αρχείο
ConfigurationLocation.txt ή επιλέξτε αρχείο παραμετροποίησης.");
                PathTxtField.setText("");
                return;
          }
          try{
                int charac = fr.read();
                String ConfigLocation ="";
                do{
                    ConfigLocation += String.valueOf((char)charac );
                    charac = fr.read();

                }while(charac != -1);
                PathTxtField.setText(ConfigLocation);
          }catch (IOException ex) {
              JOptionPane.showMessageDialog(null,"IOException κατά
την ανάγνωση του αρχείου παραμετροποίησης. Προσπαθήστε ξανά.");
              PathTxtField.setText("");
              return;
          }finally {
            try {
                fr.close();
            } catch (IOException ex) {
//                    τίποτα προς εκτέλεση.
            }
         }
    }
    public static void main(String args[]) {
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new ConvertApplicUI().setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify
    private javax.swing.JButton Browse;
    private javax.swing.JPanel ConfigPane;
    private javax.swing.JButton Exit2;
    private javax.swing.JTextField PathTxtField;
```

```
    private javax.swing.JButton StartBut;
    private javax.swing.JButton StopBut;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JPanel jPanel2;
    private javax.swing.JPanel jPanel4;
    private javax.swing.JTabbedPane jTabbedPane1;
    // End of variables declaration

}
```