Εφαρμογή Κινητών Συσκευών για την Ψηφοφορία Τηλεοπτικών Προγραμμάτων και την Παροχή Προτάσεων

Πανεπιστήμιο Πειραιώς

Σχολή Τεχνολογιών Πληροφορικής και Επικοινωνιών

Τμήμα Ψηφιακών συστημάτων

ΛΑΛΙΩΤΗΣ ΔΗΜΗΤΡΗΣ

ΜΕ13044

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Επιβλέπων καθηγητής: ΚΥΡΙΑΖΗΣ ΔΗΜΟΣΘΕΝΗΣ

Περιεχόμενα

1. ΕΙΣΑΓΩΓΗ - ΣΚΟΠΟΣ ΕΡΓΑΣΙΑΣ

Σκοπός αυτού του εγγράφου και όλης της διπλωματικής εργασίας, είναι να δείξει ένα τρόπο ώστε μία ιδέα να γίνει πραγματικότητα. Είναι δύσκολο σε μία κοινωνία όπως την δική μας, μία πρωτοποριακή ιδέα να γίνει πραγματικότητα.

Ας πάρουμε μερικά σενάρια λοιπόν, φυσικά όλα αυτά τα σενάρια θα είναι βασισμένα στην πραγματικότητα. Ας θέσουμε λοιπόν τα όρια του συστήματος (boundaries), εφόσον βρισκόμαστε στην Ελλάδα, το πιο λογικό είναι να πούμε ότι μιλάμε για την Ελλάδα. Η Ελλάδα που ξέρει και περηφανεύεται ότι έχει καλές ιδέες, αλλά με κάποιο μαγικό τρόπο που κανείς δεν προσπαθεί να αναλύσει, τίποτα δεν υλοποιείται.

Μία ιδέα για να εκφραστεί καταρχάς πρέπει να περάσει από διάφορους ανθρώπους, είτε με -σακάκια, -ράσα, -μπότες, - κ.α. είτε από την ίδια την κοινωνία που δεν καταλαβαίνει, είτε από όποιο σύστημα έχει δενδροειδή μορφή και για να φτάσει κάτι από την βάση μέχρι την κορυφή πρέπει να περάσει από πολλά επίπεδα, πολλές φορές αμφιβόλου ποιότητας. Ακόμα και αν περάσει αυτά το στάδια πρέπει να κάνει την ανάστροφη πορεία για να γίνει προϊόν που εξυπηρετεί μία ανθρώπινη ανάγκη, καθώς φυσικά ο στόχος μίας ιδέας είναι να εξυπηρετήσει μία ανθρώπινη ανάγκη, αλλιώς είναι αποτυχημένη ιδέα εξαρχής. Είτε πλέον η κορυφή είναι κάποιος που πριν ήταν στα χαμηλά στρώματα, είτε η ιδέα ήρθε από την κορυφή εξαρχής, πρέπει η βασική ιδέα να ταξιδέψει πάλι προς τα κάτω, γιατί αυτοί είναι που θα εργαστούν για να υλοποιηθεί.

Καταλαβαίνει κανείς ότι όπως και να έχει για να γίνει κάτι σε μία οποιαδήποτε χώρα, θα πρέπει να υπάρχει σωστή διακυβέρνηση. Πώς μπορείς να κάνεις λοιπόν σωστή διακυβέρνηση; Σε αυτό το σημείο θα πρέπει πάλι να θέσω τα όρια του συστήματος, άρα θα αναγκαστώ να αναλύσω περισσότερο τον σκοπό της εργασίας. Ας πούμε πως είμαστε στην διοίκηση ενός καναλιού, που θέλει να παρέχει διαφάνεια, αξιοπιστία και γιατί όχι σωστή ενημέρωση. Για να μπορέσει κάποιος να ασκήσει εξουσία, πρέπει να γνωρίζει τί συμβαίνει σε αυτό που διοικεί, αυτόν τον σκοπό αναλαμβάνει να εκπληρώσει το έργο που ακολουθεί.

2. ΤΟ ΠΡΟΒΛΗΜΑ

Όλοι γνωρίζουμε ότι η τηλεόραση είχε πάντα πρόβλημα να μεταδώσει ένα ποιοτικό προϊόν και για αυτό ο κόσμος πλέον που έχει εξελίξει το νοητικό του επίπεδο, έχει αποστραφεί τελείως από αυτό το μέσο μετάδοσης πληροφοριών και ιδεών. Πλέον, η ενημέρωση έχει βρει έναν εξελιγμένο τρόπο να διαδίδεται, το οποίο στον εικοστό πρώτο αιώνα λέγεται διαδίκτυο. Το διαδίκτυο θεωρείται όαση διοχέτευσης ιδεών, το οποίο είναι το προϊόν που προσφέρει και για αυτό χρησιμοποιείται παγκοσμίως, όποιος θέλει μπορεί να μιλήσει, μπορεί να πει την οπτική του γωνία, μπορεί να κάνει την δουλεία του σαν δημοσιογράφος, σαν καθηγητής, σαν ποδοσφαιριστής, σαν ζογκλέρ, σαν ότι θέλει να προβληθεί. Για αυτό πλέον, όλες οι ιδέες περνάνε από το διαδίκτυο.

Η τηλεόραση σαν σύστημα δεν είναι ένα κακό προϊόν από τα θεμέλια του. Έχει κάποια πράγματα να προσφέρει, όπως το έκανε τόσα χρόνια και ίσως να είναι και ένας μεγάλος λόγος που μπήκαμε τόσο γρήγορα στο διαδίκτυο. Από τότε που η ανθρωπότητα ξεκίνησε να βρίσκει τρόπους να κάνει καλύτερη την ενημέρωση, έχει μπει σε μία λογαριθμικά αύξουσα πορεία στις επιστήμες που κάνουν την ζωή μας πιο εύκολη. Αν μπορέσει λοιπόν η τηλεόραση να εργαστεί με το διαδίκτυο, σίγουρα θα κερδίσει η τηλεόραση, σίγουρα όμως και το διαδίκτυο, που και αυτό δεν είναι τέλειο, έχει τα δικά του θέματα αλλά είναι εκτός του πεδίου αυτού του εγγράφου για να αναλυθεί. Η ένωση των δύο μέσων στο εξωτερικό έχει αρχίσει να γίνεται και όντως και τα δύο κερδίζουν, στην ελληνική τηλεόραση όμως είναι πολύ πίσω οι εξελίξεις.

Παρ 'όλα αυτά, έχουν ξεκινήσει και στην χώρα μας τέτοιες κινήσεις που μπορούν να εξελιχθούν παράλληλα με τα έξω δρόμενα, αλλά όπως είπαμε οι Έλληνες έχουν καλές ιδέες και δεν συνίσταται απλά να αντιγραφούν αυτό που γίνεται στο εξωτερικό, πρέπει να υιοθετηθούν κάποια σωστά πράγματα και είτε να προσαρμοστούν στην δικιά μας κουλτούρα, είτε να διαφοροποιηθούν προς μία βελτιωμένη έκδοση.

3. ΠΡΟΤΕΙΝΟΜΕΝΗ ΛΥΣΗ

Η υλοποίηση της εργασίας αυτής είναι να συνδέσει το διαδίκτυο με την τηλεόραση. Μία αχίλλειος φτέρνα της τελευταίας, είναι ότι δεν υπάρχει ανάδραση από αυτό που παρουσιάζει. Δεν έχει κάποια είσοδο να δώσει την ανάλογη έξοδο. Όλα τα συστήματα έχουν μία είσοδο και μία έξοδο, δεν έχει ανακαλυφθεί ακόμα το σύστημα το οποίο θα σου δίνει την επιθυμητή έξοδο χωρίς να δοθεί κάποια ανάλογη είσοδο. Ίσως να μην γίνεται, αν και υπάρχουν κάποιοι επιστήμονες του μαθηματικού χώρου που κάνουν μία τέτοια έρευνα. Μέχρι να ανακαλυφθεί αυτό θα πρέπει κάποιος ο οποίος βλέπει τηλεόραση να μπορεί να κρίνει αυτό που είδε και να μπορεί να εκδηλώσει την κρίση αυτή με κάποιον τρόπο που τώρα φαντάζει απλά μία ιδέα. Το διαδίκτυο λοιπόν είναι ένας τέτοιος τρόπος.

Χρειάζεται να κατασκευαστεί μία ψηφιακή εφαρμογή που θα ενώνει αυτά τα δύο μέσα μετάδοσης. Επειδή η ενημέρωση θέλουμε να είναι γρήγορη, άρα πρέπει να είναι και άμεση, χρειάζεται μία εφαρμογή κινητού τηλεφώνου - mobile application. Είναι ένα μέσο το οποίο χρησιμοποιεί συνέχεια οποιοσδήποτε, άρα ικανοποιεί αυτά που ζητήθηκαν προηγουμένως. Η εφαρμογή πιο συγκεκριμένα δίνει την δυνατότητα στον χρήστη να κρίνει και να ψηφίσει για το αν άξιζαν οι ώρες που επένδυσε σε αυτό που είδε. Εάν και άλλοι έχουν την ίδια άποψη, τότε πριν κάποιος επόμενος θελήσει να δει μία εκπομπή και παρατηρήσει ότι δεν έχει καλή αξιολόγηση, μάλλον δεν θα ασχοληθεί και πολύ έως ότου απλά κλείσει την τηλεόραση και κάνει κάτι διαφορετικό. Έτσι η τηλεόραση δεν θα συνεχίσει να προσφέρει ανούσιο προϊόν, αλλά θα δείχνει κάτι το οποίο έχει αξιολογηθεί από το κοινό του.

3.1 Προδιαγραφές & περιγραφή λειτουργικότητας

Η εφαρμογή λοιπόν επειδή έχει σκοπό να καλύψει καθολικά το μέσο - τηλεόραση - και όχι κάποιο συγκεκριμένο κανάλι, δίνει την δυνατότητα στον χρήστη να επιλέξει το κανάλι που θέλει να βρει μια εκπομπή, να δει το πρόγραμμα του καναλιού, να επιλέξει την εκπομπή που θέλει να αξιολογήσει και εν τέλει να δώσει την ψήφο του, η οποία θα είναι ένας αριθμός από το 0 έως το 5. Στον χρήστη θα φαίνονται πόσα αστέρια βάζει, αλλά στην ουσία ένας αριθμός είναι αυτό που χρειάζεται. Κάπως έτσι το πρόβλημα παίρνει μαθηματική υπόσταση, το άθροισμα των αξιολογήσεων δια τον αριθμό των ατόμων που ψήφισαν είναι ο μέσος όρος της ψηφοφορίας, ένα νούμερο που θα

χρησιμοποιηθεί για την κατάταξη της κάθε εκπομπής. Έπειτα, εμφανίζονται οι προτεινόμενες εκπομπές με βάση τις επιλογές του χρήστη.

Χ = Μέσος όρος ψήφων

Ψ = Αριθμός ψήφων

Ω = Άθροισμα ψήφων

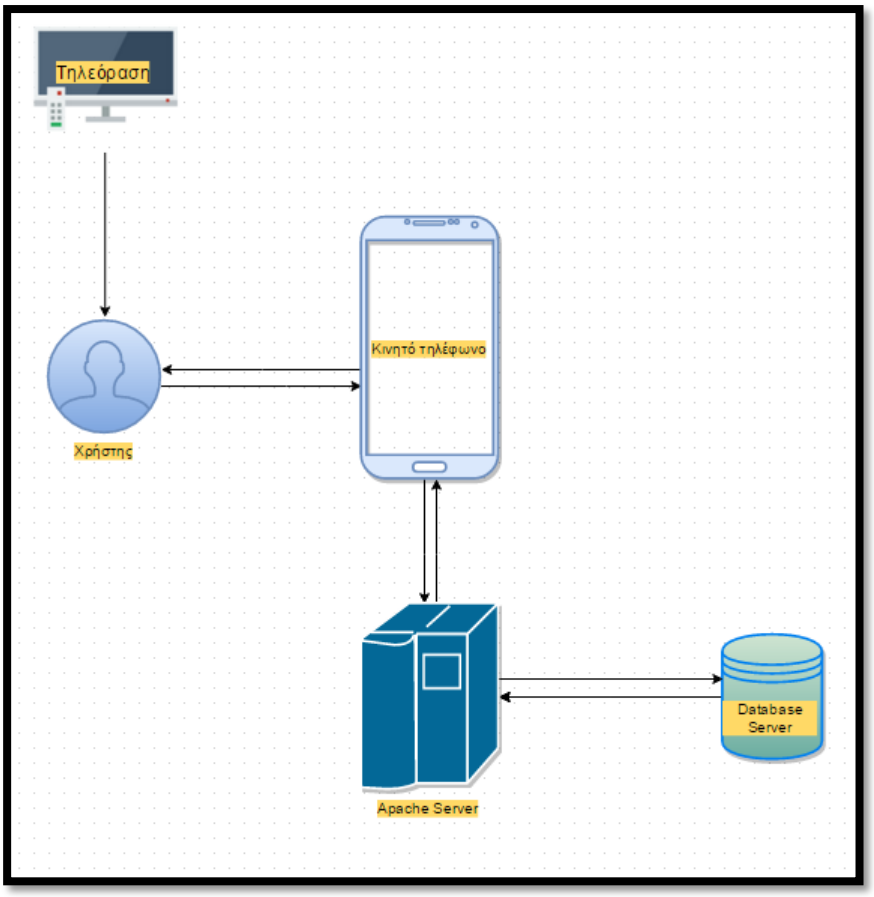
Χ = Ω / Ψ

Τέλος, αφού ο χρήστης δώσει την ψήφο του, η εφαρμογή θα προτείνει τρείς παρόμοιες εκπομπές. Ο τρόπος της εκλογής αυτής θα γίνεται με τον αλγόριθμο συνεργατικού φιλτραρίσματος. Περιληπτικά, ο αλγόριθμος υπολογίζει τις προτιμήσεις των υπόλοιπων χρηστών, βρίσκει χρήστες με παρόμοιες απόψεις και επιστρέφει τις τρείς καλύτερες που ανήκουν στην ίδια κατηγορία με την εκπομπή που μόλις άσκησε κριτική ο τρέχων χρήστης.

Για το έργο αυτό θα χρησιμοποιηθεί το android studio, οπότε απευθύνεται μόνο σε χρήστες android.

4. ΣΧΕΤΙΚΕΣ ΤΕΧΝΟΛΟΓΙΕΣ

Η αρχιτεκτονική του συστήματος θα είναι πολύ απλή για τα σημερινά δεδομένα. Η τηλεόραση δεν συμμετέχει άμεσα στο σύστημα αλλά είναι ένα κομμάτι του. Έχουμε λοιπόν την εφαρμογή στην συσκευή του κινητού οπού ο χρήστης δίνει την άποψή του, για την εφαρμογή είσοδος του συστήματος. Στέλνει δεδομένα σε ένα Server που τα κρατάει και κάνει τους υπολογισμούς για τα στατιστικά. Η εφαρμογή στο κινητό είναι φτιαγμένη με το Android studio, άρα αναπτυγμένη σε Java. Ο Server αποτελείται από 2 κομμάτια, την βάση δεδομένων που κρατάει τα δεδομένα και έναν Apache Server ο οποίος έχει τον ρόλο του διαμεσολαβητή ανάμεσα στο κινητό και την βάση.



Εικόνα 1 - Αρχιτεκτονική τεχνικών υποδομών

4.1 Android SDK[1]

Η ανάπτυξη λογισμικού Android είναι η διαδικασία κατα την οποία δημιουργείται μία εφαρμογή για το λειτουργικό σύστημα του Android. Οι εφαρμογές αυτές, συνήθως αναπτύσσονται σε προγραμματιστική γλώσσα Java χρησιμοποιώντας το Adroid Software Development Kit (SDK), υπάρχουν διαθέσιμα όμως και άλλα περιβάλλοντα ανάπτυξης.

Απο τον ιούλιο του 2013, περισσότερο απο ένα εκκατομύριο εφαρμογες έχουν δημιουργηθεί για το Android, με παραπάνω απο 25 εκκατομύρια κατευάσματα. Το Ιούνιο του 2011 μία έρευνα έδειξε οτι πάνω απο το 67% των προγραμματιστών για κινητές συσκευές χρησιμοποιούσαν αυτήν την πλατφόρμα κατα την εποχή της δημοσιοποίησης αυτής της έρευνας. Το δεύτερο τέταρτο του 2012, περίπου 105

εκκατομύρια μονάδες έξυπνων κινητών τηλεφώνων Android πωλήθηκαν, ένα νούμερο το οποίο καλύπτει το 68% των πωλήσεων στην αγορά έξυπνων κινητών τηλεφώνων.

Το Android software development kit (SDK) περιλαμβάνει ένα περιεκτικό σύνολο από εργαλεία ανάπτυξης. Αυτά αποτελούνται από debuggers, βιβλιοθήκες, emulators βασισμένοι στο QEMU, έγγραφα για βοήθεια, δείγματα κώδικα και εκπαιδευτικά προγράμματα. Αυτήν την στιγμή, υποστηρίζεται από πλατφόρμες με λειτουργικό σύστημα Linux, Mac OS X 10.5.8 ή ύστερο, Windows XP ή ύστερο, Έως τον Μάρτιο του 2015, το SDK δεν είναι διαθέσιμο για το ίδιο το Android, αλλά η ανάπτυξη λογισμικού είναι δυνατή με την χρήση εξειδικευμένων εφαρμογών Android.

Μέχρι περίπου το τέλος του 2014, το ολοκληρωμένο περιβάλλον ανάπτυξης (Intergrated Development Environment - IDE) που υποστηριζόταν ήταν το Eclipse, το οποίο χρησιμοποιούσε τα εργαλεία Android Development Tools (ADT), μολονότι πάντα υπήρχε το IntelliJ IDEA IDE [2](όλες οι εκδόσεις του), το οποίο υποστήριζε και υποστηρίζει ακόμα την πλήρη ανάπτυξη κώδικα για Android από την αρχή, ήδη υπάρχουν και άλλα εργαλεία ανάπτυξης για Android, όπως το NetBeans IDE το οποίο υποστηρίζει πλήρως την Android πλατφόρμα.

Το Android Studio, το οποίο δημιουργήθηκε από την Google εν δυνάμει IntelliJ, είναι το πλέον επίσημο εργαλείο ανάπτυξης για Android (Android IDE)· παρόλο πού οι προγραμματιστές είναι ελεύθεροι να χρησιμοποιήσουν όποιο άλλο θέλουν. Πλέον, οι προγραμματιστές μπορούν να χρησιμοποιήσουν οποιοδήποτε πρόγραμμα επεξεργασίας κειμένου για να επεξεργαστούν τα Java και XML αρχεία, μετά μπορούν να δημιουργήσουν έργα με εργαλεία εκτέλεσης μέσα από γραμμές εντολών (Τα Java Development Kit και Apache Ant είναι απαραίτητα για να πραγματοποιηθούν αυτά), να χτίσουν, να δημιουργήσουν, να κάνουν ελέγχους της τελικής Android εφαρμογής. Όλα αυτά μέσα από το *Android Studio* της *Google*.

Η εξέλιξη του εργαλείου ανάπτυξης Android Studio συμβαδίζει με την συνολική ανάπτυξη της πλατφόρμας του Android. Επίσης υποστηρίζει παλιότερες εκδόσεις της Android πλατφόρμας σε περίπτωση που οι προγραμματιστές θέλουν να στοχεύσουν την εφαρμογή τους σε παλιότερες συσκευές που πιθανώς να έχει ο κόσμος. Όλα αυτά είναι ελεύθερα διαθέσιμα στο διαδίκτυο στις τελευταίες εκδόσεις τους πάντα, χωρίς κανένα κόστος.

Οι Android εφαρμογές πακετάρονται σε μορφή .apk και αποθηκεύονται στον φάκελο με διεύθυνση /data/app στο λειτουργικό σύστημα (ο φάκελος είναι διαθέσιμος μόνο για

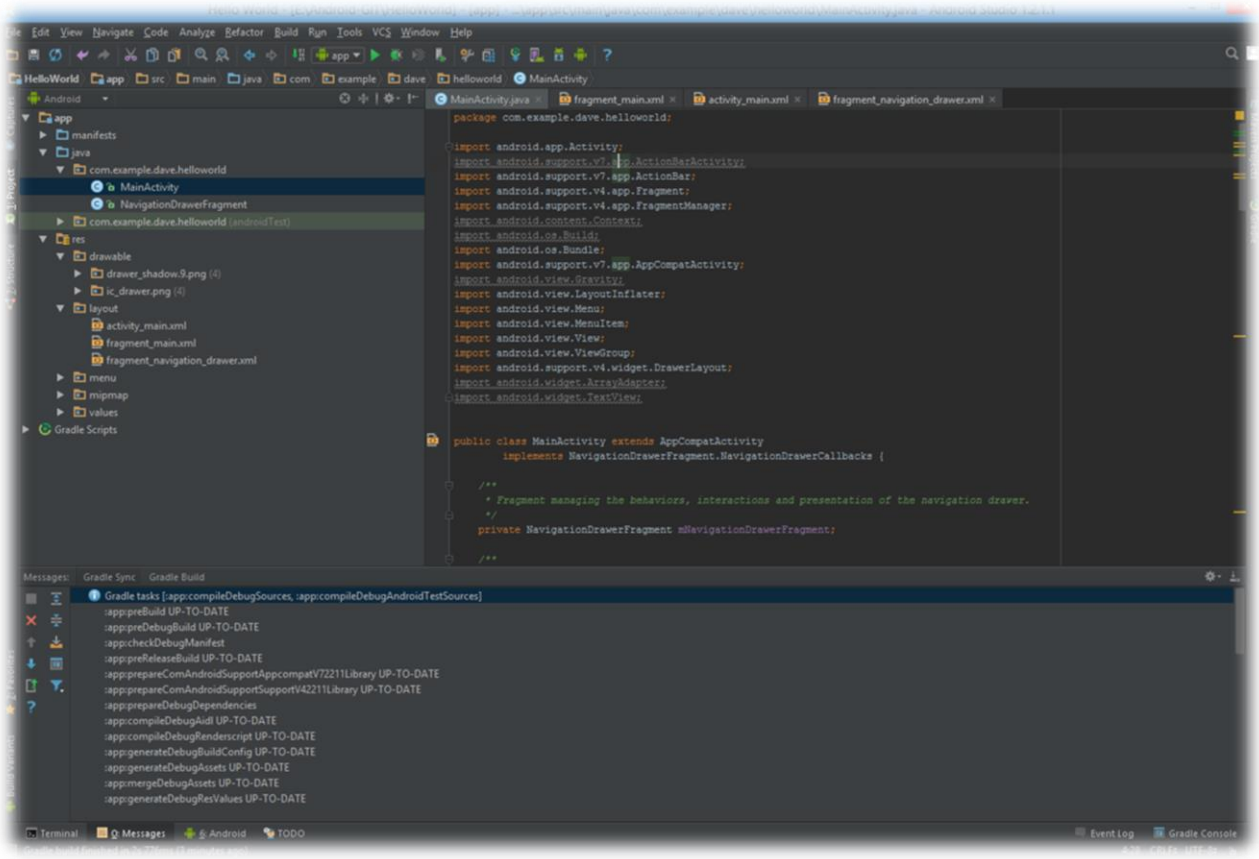
τον ρόλο του διαχειριστή). Το πακέτο APK περιέχει .dex αρχεία (μεταγλωττισμένος κώδικας χαμηλού επιπέδου εν ονόματι Dalvik executables), αρχεία πόρων (resources files), κ.λπ.

4.1.1 Android Studio

Το Android Studio είναι ένα ολοκληρωμένο προγραμματιστικό περιβάλλον για την ανάπτυξη εφαρμογών Android. Ανακοινώθηκε στης 16 Μαΐου του 2013 στην ανοιχτή συνδιάσκεψη της google απο την διευθύντρια του προιόντος, Katherine Chou. Το Android Studio είναι ελεύθερα διαθέσιμο υπο την άδεια της Apache.

Επίσημο σήμα - Android Studio

Το Android Studio ήταν ήδη στο πρώιμο στάδιο στην έκδοση 0.1 τον Μάιο του 2013, μετά μπήκε στο στάδιο βήτα (beta) ξεκινώντας από την έκδοση 0.8 η οποία κυκλοφόρησε τον Ιούνιο του 2014. Η πιο σταθερή έκδοση κυκλοφόρησε τον Δεκέμβριο του 2014, ξεκινώντας από την έκδοση 1.0.



Εικόνα 2 - Στιγμιότυπο του Android Studio κατά την διάρκεια ανάπτυξης του Tv Vote Cast

Βασισμένο στο λογισμικό IntelliJ IDEA της JetBrains, είναι σχεδιασμένο ειδικά για ανάπτυξη εφαρμογών Android. Είναι διαθέσιμο για κατέβασμα για Windows, Mac OS x και Linux, και αντικαθυστά το Eclipse με τα εργαλεία ανάπτυξης Android (Android Development Tools - ADT), γιατί είναι πλέον το βασικό IDE για εγγενή Android εφαρμογές της Google.

4.1.2 Ιστορική αναδρομή android

Το Android δημιουργήθηκε απο την Open Handset Alliance, η οποία ηγείται από την Google. Η πρόβλεψη για το μέλλον της Android πλατφόρμας στα πρώιμα στάδια ήταν ανάμεικτη και το κλίμα ακόμα αβέβαιο. Υπήρχαν προγραμματιστικά προβλήματα (bugs), έλλειψη περιγραφικών εγγράφων, ανεπαρκής υποδομές συστημάτων για ερωτήσεις και απαντήσεις σχετικά με την πλατφόρμα, χωρίς κάποιο σύστημα δημόσιας παρακολούθησης θεμάτων. Η Google μόλις στις 18 Ιανουαρίου του 2008 ανακοίνωσε την ύπαρξη ενός τέτοιου συστήματος, που παρακολουθεί την κίνηση της Android

πλατφόρμας, δηλώνει με περιγραφικά έγγραφα όλες τις δυνατότητες και πώς μπορούν να αξιοποιηθούν, και τέλος απαντάει σε ερωτήσεις των προγραμματιστών, την ίδια την κοινότητα που χρησιμοποιεί και δουλεύει με το Android. Τον Δεκέμβριο του 2007, ο ιδρυτής της νεοσύστατης MergLab, MacBeth δήλωσε (σ.σ. ελεύθερη μετάφραση) "Λειτουργικότητα δεν υπάρχει ακόμα, ή είναι κακό ξηγημένο ή απλά δεν δουλεύει... Είναι σαφές ότι δεν είναι έτοιμο ακόμα". Παρά το γεγονός αυτό, εφαρμογές που είχαν στόχο το Android ξεκίνησαν να εμφανίζονται την επομένη της εβδομάδας που ανακοινώθηκε η κυκλοφορία του. Το Android Dev Phone είναι μία συσκευή που έχει ξεκλείδωτες τις λειτουργίες της SIM και ξεκλείδωτες όλες τις λειτουργίες προς το υλικό (Hardware), για προχωρημένους προγραμματιστές. Την στιγμή που άλλοι προγραμματιστές μπορούν να χρησιμοποιήσουν κοινές καταναλωτικές συσκευές του εμπορίου, για να ελέγξουν, να δοκιμάσουν, να χρησιμοποιήσουν τις εφαρμογές τους. Είναι όλα απαραίτητα βήματα για την υλοποίηση του αποτελέσματος.

Μία έκδοση προεπισκόπησης του Android SDK κυκλοφόρησε τον Νοέμβριο του 2007. Τον Ιούλιο του 2008, μία ομάδα δημιουργών του Android έστειλε ένα ηλεκτρονικό γράμμα κατά λάθος σε όλους τους συμμετέχοντες, ανακοινώνοντας ότι μια νέα έκδοση του SDK είναι διαθέσιμη για κατέβασμα από έναν ιδιωτικό χώρο. Το γράμμα προοριζόταν για τους νικητές του πρώτου γύρου του διαγωνισμού "Android Developer Challenge", η αποκάλυψη ότι η Google παρείχε το καινούριο SDK μόνο σε μία μερίδα προγραμματιστών και όχι σε όλους, κρατώντας και την συμφωνία μυστική, οδήγησε σε μία ευρύ απογοήτευση εντός της κοινότητας των Android προγραμματιστών εκείνη την περίοδο.

Τον Αύγουστο το 2008, κυκλοφόρησε το Android 0.9 SDK beta. Αυτή η κυκλοφορία παρείχε ένα αναβαθμισμένο και εκτενέστερο API (Application Programming Interface), ένα αναβαθμισμένο εργαλείο και ένα αναβαθμισμένο γραφικό σχέδιο στην αρχική οθόνη. Λεπτομερείς οδηγίες για τις καινούριες αναβαθμίσεις είναι διαθέσιμες σε όσους έχουν ήδη δουλέψει σε παλιότερες εκδόσεις. Τον Σεπτέμβριο του 2008, το Android 1.0 SDK (Release 1) ήρθε στην κυκλοφορία. Σύμφωνα με τις σημειώσεις της έκδοσης, περιελάμβανε κυρίως διορθώσεις προγραμματιστικών λαθών, αν και προσθέθηκαν μερικά μικρά καινούρια χαρακτηριστικά. Επίσης περιείχε αλλαγές στο API σε σχέση με αυτό της έκδοσης 0.9. Απο τότε έχουν κυκλοφορίσει πολλές καινούριες εκδόσεις, καταλείγοντας στο σήμερα -αρχές 2016- που έχουμε φτάσει στο Android 6.0 Marshmallow.

4.2 Java[3]

Η Java είναι γλώσσα προγραμματισμού που σχεδιάστηκε από την εταιρεία πληροφορικής Sun Microsystems. Είναι ταυτόχρονα μία γλώσσα προγραμματισμού γενικού σκοπού, αντικειμενοστραφής, βασίζεται σε κλάσεις, και είναι σχεδιασμένη να έχει όσο το δυνατόν λιγότερες εξαρτήσεις(dependencies) από τρίτες βιβλιοθήκες. Σκοπεύει να δώσει την δυνατότητα στον προγραμματιστή να γράψει μία φορά και ο κώδικας να εκτελείται παντού, επειδή ο Java κώδικας που έχει μεταγλωττιστεί, εκτελείται σε όλες τις πλατφόρμες που την υποστηρίζουν χωρίς να χρειάζεται επαναμεταγλώττιση. Οι εφαρμογές Java είναι τυπικά μεταγλωττισμένες σε δυαδικός κώδικα που υποστηρίζεται από οποιοδήποτε εικονικό μηχάνημα Java (Java virtual machine - JVM) ανεξαρτήτως αρχιτεκτονικής του υπολογιστή. Το 2015, η Java είναι μία από τις πιο γνωστές γλώσσες προγραμματισμού σε χρήση, ιδιαίτερα για client-server δικτυακές εφαρμογές, με ένα πλήθος 9 εκατομμυρίων δηλωμένων προγραμματιστών. Η Java αρχικά δημιουργήθηκε από τον James Gosling στην εταιρεία Sun Microsystems (η οποία μετέπειτα εξαγοράστηκε από την Oracle) και το 1995 κυκλοφόρησε ως βασικό της συστατικό την Java πλατφόρμα. Η σύνταξη της γλώσσας προέρχεται αρκετά από την C και την C++, αλλά έχει λιγότερες λειτουργίες χαμηλού επιπέδου από αυτές.

Επίσημο σήμα - Java

Οι πρωτότυποι μεταγλωττιστές Java, οι εικονικές μηχανές και οι βιβλιοθήκες με τις κλάσεις της Java εκδόθηκαν κάτω από ιδιόκτητες άδειες της Sun. Τον Μάιο του 2007, σε συνεργασία με τις προδιαγραφές της κοινότητας της Java, η Sun επανέκδωσε τις περισσότερες Java τεχνολογίες κάτω από την άδεια GNU (General Public License). Άλλοι έχουν επίσης αναπτύξει εναλλακτικές εφαρμογές των τεχνολογιών της Sun, όπως ο GNU μεταγλωττιστής για την Java, GNU Classpath, και IcedTea-Web.

Η τελευταία έκδοση είναι η Java 8, η μόνη έκδοση που υποστηρίζεται αυτήν την στιγμή.

4.2.1 Τα χαρακτηριστικά της Java

Ένα από τα βασικά πλεονεκτήματα της Java έναντι των περισσότερων άλλων γλωσσών είναι η ανεξαρτησία του λειτουργικού συστήματος και πλατφόρμας. Τα προγράμματα που είναι γραμμένα σε *Java* τρέχουνε ακριβώς το ίδιο σε Windows, Linux, Unix και Macintosh (σύντομα θα τρέχουν και σε Playstation καθώς και σε άλλες κονσόλες παιχνιδιών) χωρίς να χρειαστεί να ξαναγίνει μεταγλώττιση (compiling) ή να αλλάξει ο πηγαίος κώδικας για κάθε διαφορετικό λειτουργικό σύστημα. Για να επιτευχθεί όμως αυτό χρειαζόταν κάποιος τρόπος έτσι ώστε τα προγράμματα γραμμένα σε Java να μπορούν να είναι «κατανοητά» από κάθε υπολογιστή ανεξάρτητα του είδους επεξεργαστή (Intel x86, IBM, Sun SPARC, Motorola) αλλά και λειτουργικού συστήματος (Windows, Unix, Linux, BSD, MacOS). Ο λόγος είναι ότι κάθε κεντρική μονάδα επεξεργασίας κατανοεί διαφορετικό κώδικα μηχανής. Οσυμβολικός κώδικας (*assembly*) που μεταφράζεται και εκτελείται σε Windows είναι διαφορετικός από αυτόν που μεταφράζεται και εκτελείται σε έναν υπολογιστή Macintosh. Η λύση δόθηκε με την ανάπτυξη της *Εικονικής Μηχανής* (*Virtual Machine* ή VM ή ΕΜ στα ελληνικά).

Η εικονική μηχανή της Java[4]

Αφού γραφεί κάποιο πρόγραμμα σε Java, στη συνέχεια μεταγλωτίζεται μέσω του μεταγλωττιστή javac, ο οποίος παράγει έναν αριθμό από αρχεία .class (κώδικας byte ή bytecode). Ο κώδικας byte είναι η μορφή που παίρνει ο πηγαίος κώδικας της Java όταν μεταγλωττιστεί. Όταν πρόκειται να εκτελεστεί η εφαρμογή σε ένα μηχάνημα, το Java Virtual Machine που πρέπει να είναι εγκατεστημένο σε αυτό θα αναλάβει να διαβάσει τα αρχεία .class. Στη συνέχεια τα μεταφράζει σε γλώσσα μηχανής που να υποστηρίζεται από το λειτουργικό σύστημα και τον επεξεργαστή, έτσι ώστε να εκτελεστεί (αυτό συμβαίνει με την παραδοσιακή Εικονική Μηχανή (Virtual Machine). Πιο σύγχρονες εφαρμογές της εικονικής Μηχανής μπορούν και μεταγλωττίζουν εκ των προτέρων τμήματα bytecode απευθείας σε κώδικα μηχανής (εγγενή κώδικα ή native code) με αποτέλεσμα να βελτιώνεται η ταχύτητα). Χωρίς αυτό δε θα ήταν δυνατή η εκτέλεση λογισμικού γραμμένου σε Java. Η JVM είναι λογισμικό που εξαρτάται από την πλατφόρμα, δηλαδή για κάθε είδος λειτουργικού συστήματος και αρχιτεκτονικής επεξεργαστή υπάρχει διαφορετική έκδοση του. Έτσι υπάρχουν διαφορετικές JVM για Windows, Linux, Unix, Macintosh, κινητά τηλέφωνα, παιχνιδομηχανές κλπ. Οτιδήποτε θέλει να κάνει ο προγραμματιστής (ή ο χρήστης) γίνεται μέσω της εικονικής μηχανής. Αυτό βοηθάει στο να υπάρχει μεγαλύτερη ασφάλεια στο σύστημα γιατί η εικονική μηχανή είναι υπεύθυνη για την επικοινωνία χρήστη – υπολογιστή. Ο προγραμματιστής δεν μπορεί να γράψει κώδικα ο οποίος θα έχει καταστροφικά αποτελέσματα για τον υπολογιστή γιατί η εικονική μηχανή θα τον ανιχνεύσει και δε θα επιτρέψει να εκτελεστεί. Από την άλλη μεριά ούτε ο χρήστης μπορεί να κατεβάσει «κακό» κώδικα από το δίκτυο και να τον εκτελέσει. Αυτό είναι ιδιαίτερα χρήσιμο για μεγάλα κατανεμημένα συστήματα όπου πολλοί χρήστες χρησιμοποιούν το ίδιο πρόγραμμα συγχρόνως.

Ο συλλέκτης απορριμάτων (Garbage Collector)[5]

Ακόμα μία ιδέα που βρίσκεται πίσω από τη *Java* είναι η ύπαρξη του συλλέκτη απορριμμάτων (*Garbage Collector*). Συλλογή απορριμμάτων είναι μία κοινή ονομασία που χρησιμοποιείται στον τομέα της πληροφορικής για να δηλώσει την ελευθέρωση τμημάτων μνήμης από δεδομένα που δε χρειάζονται και δε χρησιμοποιούνται άλλο. Αυτή η απελευθέρωση μνήμης στη Java είναι αυτόματη και γίνεται μέσω του συλλέκτη απορριμμάτων. Υπεύθυνη για αυτό είναι και πάλι η εικονική μηχανή η οποία μόλις «καταλάβει» ότι ο σωρός (heap) της μνήμης (στη Java η συντριπτική πλειοψηφία των αντικειμένων αποθηκεύονται στο σωρό σε αντίθεση με τη C++ όπου αποθηκεύονται κυρίως στη στοίβα) κοντεύει να γεμίσει ενεργοποιεί το συλλέκτη απορριμμάτων. Έτσι ο προγραμματιστής δε χρειάζεται να ανησυχεί για το πότε και αν θα ελευθερώσει ένα

συγκεκριμένο τμήμα της μνήμης, ούτε και για σφάλματα δεικτών. Αυτό είναι ιδιαίτερα σημαντικό γιατί είναι κοινά τα σφάλματα προγραμμάτων που οφείλονται σε λανθασμένο χειρισμό της μνήμης.

Επιδόσεις

Παρόλο που η εικονική μηχανή προσφέρει όλα αυτά (και όχι μόνο) τα πλεονεκτήματα, η Java αρχικά ήταν πιο αργή σε σχέση με άλλες προγραμματιστικές γλώσσες υψηλού επιπέδου (high-level) όπως η C και η C++. Εμπειρικές μετρήσεις στο παρελθόν είχαν δείξει ότι η C++ μπορούσε να είναι αρκετές φορές γρηγορότερη από την Java. Ωστόσο γίνονται προσπάθειες από τη Sun για τη βελτιστοποίηση της εικονικής μηχανής, ενώ υπάρχουν και άλλες υλοποιήσεις της εικονικής μηχανής από διάφορες εταιρίες (όπως της IBM), οι οποίες μπορεί σε κάποια σημεία να προσφέρουν καλύτερα και σε κάποια άλλα χειρότερα αποτελέσματα. Επιπλέον με την καθιέρωση των μεταγλωττιστών JIT (Just In Time), οι οποίοι μετατρέπουν τον κώδικα byte απευθείας σε γλώσσα μηχανής, η διαφορά ταχύτητας από τη C++ έχει μικρύνει κατά πολύ.
Οι τελευταίες εκδόσεις του javac με τη χρήση της τεχνολογίας Hot Spot έχουν καταφέρει αξιόλογες επιδόσεις που πλησιάζουν ή και ξεπερνούν σε μερικές περιπτώσεις τον εγγενή κώδικα.

4.2.2 Μεταγλώττιση εκείνη την στιγμή (JIT)[6]

Ο just-in-time (JIT) μεταγλωττιστής είναι η καρδιά της Java εικονικής μηχανής. Τίποτα δεν επηρεάζει το εικονικό μηχάνημα περισσότερο από τον μεταγλωττιστή, και για αυτόν τον λόγο η επιλογή του μεταγλωττιστή είναι μία από τις πρώτες αποφάσεις που πρέπει να πάρουν οι προγραμματιστές, όταν δημιουργούν μία εφαρμογή Java. Ευτυχώς, στις περισσότερες περιπτώσεις ο μεταγλωττιστής χρειάζεται μόνο κάποιες βασικές ρυθμίσεις για να λειτουργήσει.

Οι υπολογιστές και πιο συγκεκριμένα οι επεξεργαστές, μπορούν να εκτελέσουν λίγες εντολές· ειδικές οδηγίες, που ονομάζονται δυαδικός κώδικας ή assembly. Όλα τα προγράμματα που θέλουν να χρησιμοποιήσουν τον επεξεργαστή πρέπει να μιλάνε αυτήν την γλώσσα, ως εκ τούτου πρέπει να μεταφράσουν τις εντολές του προγραμματιστή στην γλώσσα του επεξεργαστή.

Γλώσσες όπως η C++ και η Fortran ονομάζονται καταρτισμένες (compliled) γλώσσες, επειδή τα προγράμματα που παράγουν, παραδίδονται σε δυαδικό κώδικα: γράφεται το πρόγραμμα και ο στατικός μεταγλωττιστής παράγει το δυαδικό. Ο assembly κώδικας σε δυαδικό έχει στόχο έναν συγκεκριμένο επεξεργαστή. Συμπληρωματικοί επεξεργαστές μπορούν να εκτελέσουν το ίδιο δυαδικό: π.χ. οι επεξεργαστές AMD και Intel μοιράζονται βασικές κοινές εντολές, και όλες οι τελευταίες εκδόσεις των επεξεργαστών αυτών μπορούν κατά ένα μεγάλο ποσοστό να εκτελέσουν τις ίδιες εντολές παλιότερων εκδόσεων. Το ανάστροφο δεν είναι πάντα αληθές, καινούριες εκδόσεις των επεξεργαστών συχνά εισάγουν εντολές που δεν μπορούν να εκτελεστούν σε παλιότερες εκδόσεις.

Από την άλλη μεριά, γλώσσες όπως η PHP και η Perl είναι διερμηνευμένες. Ο ίδιος πηγαίος κώδικας εκτελείται σε οποιονδήποτε επεξεργαστή από την στιγμή που το μηχάνημα έχει τον σωστό διερμηνευτή (αυτό είναι το πρόγραμμα εν ονόματι Php ή Perl). Ο διερμηνευτής μεταφράζει την κάθε γραμμή του προγράμματος σε δυαδικό κώδικα και την εκτελεί.

Υπάρχουν πλεονεκτήματα και μειονεκτήματα για το κάθε σύστημα. Προγράμματα σε γλώσσα Php ή Perl είναι φορητά, ωστόσο μπορεί να έχουν μεγάλο χρόνο εκτέλεσης. Σε ένα απλό σενάριο επαναληπτικής λειτουργίας, ο διερμηνευτής θα ξανά μεταφράσει κάθε γραμμή κώδικα για κάθε βρόγχο. Ο κώδικας ο οποίος είναι μεταγλωττισμένος δεν χρειάζεται να κάνει αυτήν την επαναλαμβανόμενη μετάφραση.

Υπάρχει ένας αριθμός από παράγοντες που ένας καλός μεταγλωττιστής λαμβάνει υπόψη όταν παράγει το δυαδικό. Ένα απλό παράδειγμα είναι η σειρά των δυαδικών δηλώσεων, ο χρόνος εκτέλεσης των εντολών σε επίπεδο assembly δεν χρειάζονται τον ίδιο χρόνο εκτέλεσης. Η εντολή αθροίσματος δύο τιμών που είναι αποθηκευμένες σε δύο καταχωρητές μπορεί να εκτελεστεί σε έναν κύκλο μηχανής, αλλά η ανάκτηση των τιμών αυτών από την μνήμη μπορεί να χρειαστεί πολλαπλούς κύκλους μηχανής.

Ως εκ τούτου, ένας καλός μεταγλωττιστής θα παράγει τον δυαδικό κώδικα που εκτελείται για την ανάκτηση των δεδομένων, παράλληλα τον δυαδικό κώδικα που εκτελείται για την επεξεργασία των δεδομένων, και μετά όταν τα δεδομένα είναι διαθέσιμα, εκτελεί την πρόσθεση. Ένας διερμηνευτής ο οποίος κοιτάει μόνο μία γραμμή κώδικα κάθε φορά δεν έχει αρκετές πληροφορίες ώστε να παράγει τέτοιον κώδικα, θα ζητήσει τα δεδομένα από την μνήμη, θα περιμένει μέχρι να του γίνουν διαθέσιμα και μετά θα πραγματοποιήσει την πρόσθεση. Παρεμπιπτόντως, ένας κακός μεταγλωττιστής θα κάνει το ίδιο αν ο πηγαίος κώδικας είναι εσφαλμένα δομημένος, σε κάποιες

περιπτώσεις ακόμα και οι καλύτεροι μεταγλωττιστές δεν μπορούν να αποφύγουν την αναμονή περάτωσης μίας εντολής.

Για αυτούς και άλλους λόγους, γλώσσες όπως η Php και Perl σχεδόν πάντα θα είναι αισθητά πιο αργές από γλώσσες που μεταγλωττίζουν τον κώδικα (Java, C#). Οι μεταγλωττιστές έχουν αρκετές πληροφορίες για το πρόγραμμα ώστε να παράγουν έναν σημαντικό πλήθος βελτιστοποιήσεων στον δυαδικό κώδικα που θα παράγουν, κάτι το οποίο ένας διερμηνευτής δεν μπορεί απλά να πραγματοποιήσει.

Κώδικας ο οποίος προέρχεται από διερμηνευτή δεν έχει το πλεονέκτημα της φορητότητας. Το δυαδικό το οποίο μεταγλωττίστηκε για έναν επεξεργαστή SPARC προφανώς δεν μπορεί να λειτουργήσει σε επεξεργαστή Intel, από την άλλη μεριά, ούτε το δυαδικό το οποίο χρησιμοποιεί τις εντολές του πιο πρόσφατου AVX της Intel δεν μπορεί να λειτουργήσει σε έναν παλιό Intel επεξεργαστή. Ως εκ τούτου, συνηθίζεται σε λογισμικά του εμπορίου να χρησιμοποιούνται παλιότερες εκδόσεις του επεξεργαστή χωρίς να είναι διαθέσιμες οι πιο καινούριες εντολές από την μία, από την άλλη όμως είναι πιο σταθερά. Υπάρχουν διάφορα κόλπα γύρω από αυτήν την φιλοσοφία, μερικά από αυτά είναι η δημιουργία έξυπνων βιβλιοθηκών που εκτελούν ευαίσθητο κώδικα σε ένα μεγάλο φάσμα διαφορετικών επεξεργαστών.

Το υλικό το οποίο εκτελείται ο κώδικας είναι το μεγαλύτερο πρόβλημα της εποχής για έναν προγραμματιστή, είτε αναφερόμαστε σε προσωπικούς υπολογιστές, είτε σε ένα μεγάλο υπολογιστικό σύστημα, είτε σε ένα κινητό τηλέφωνο, τα κομμάτια τα οποία απαρτίζουν μία τέτοια συσκευή και οι συνδυασμοί αυτών είναι πάρα πολλοί. Ένας λόγος που μία εφαρμογή ή ένα πρόγραμμα αποτυγχάνει να εκτελεστεί είναι ότι το υλικό δεν μπορεί να υπακούσει τις εντολές του προγραμματιστή επειδή του είναι άγνωστες.

Η Java προσπαθεί να βρει μία μέση λύση στο πρόβλημα αυτό, οι εφαρμογές της μεταγλωττίζονται, αλλά αντί να μεταγλωττίζονται σε συγκεκριμένο δυαδικό για συγκεκριμένο επεξεργαστή, μεταγλωττίζονται σε μία εξιδανικευμένη γλώσσα assembly. Η γλώσσα αυτή, γνωστή ως Java bytecode, στην συνέχεια εκτελείται από το δυαδικό της Java με παρόμοιο τρόπο όπως θα έτρεχε ο κώδικας Php από ένα Php δυαδικό. Αυτό δίνει στην Java ανεξαρτησία σε σχέση με την πλατφόρμα, όπως έχουν γλώσσες που χρησιμοποιούν διερμηνευτή. Επειδή εκτελεί εξιδανικευμένο δυαδικό κώδικα, οι βιβλιοθήκες της Java είναι ικανές να μεταγλωττίσουν τον κώδικα σε δυαδικό κώδικα που αρμόζει στην πλατφόρμα. Αυτή η μεταγλώττιση συμβαίνει όταν το πρόγραμμα εκτελείται· συμβαίνει "εκείνη την στιγμή".

Παράλληλη μεταγλώττιση

Ο μεταγλωττιστής κατά την διάρκεια της εκτέλεσης δημιουργεί μία ουρά από μεθόδους και δομές επανάληψης για να τις μεταγλωττίσει. Η ουρά επεξεργάζεται από ένα ή περισσότερα νήματα στο παρασκήνιο, αυτό σημαίνει ότι η μεταγλώττιση είναι μία ασύγχρονη διαδικασία· κάτι το οποίο είναι καλό. Επιτρέπει στο πρόγραμμα να συνεχίσει να εκτελείται, ακόμα και όταν ο κώδικας που ακολουθεί ακόμα είναι υπό μεταγλώττιση.

Η σειρά μεταγλώττισης δεν ακολουθείται πιστά με την λογική του ότι μπαίνει πρώτο βγαίνει και πρώτο. Μέθοδοι οι οποίες έχουν μεγαλύτερη προτεραιότητα θα έχουν και μεγαλύτερη προτεραιότητα στην ουρά. Έτσι λοιπόν, όταν ένα πρόγραμμα ξεκινήσει να εκτελείται και έχει πολύ κώδικα να μεταγλωττίσει, η ανάθεση προτεραιοτήτων είναι σημαντικός παράγοντας για την ταχύτητα του προγράμματος.

Ενσωμάτωση

Μία από τις σημαντικότερες βελτιστοποιήσεις του μεταγλωττιστή είναι η μέθοδος της ενσωμάτωσης. Κώδικας ο οποίος ακολουθεί ένα καλό σχέδιο αντικειμενοστραφή προγραμματισμού περιέχει ένα μεγάλο αριθμό ορισμάτων τα οποία προσπελαύνονται από getters **και** setters.

```
public class Point
{
    private int x, y;

    public void getX()
    {
        return x;
    }

    public void setX(int i)
    {
        x = i;
    }
}
```

Το κόστος επίκλησης μεθόδου σαν και αυτή είναι πολύ υψηλό, ειδικά αν αναλογιστεί κανείς την ποσότητα του κώδικα που εκτελείται. Σε αρχικά στάδια της Java, συμβουλές για καλύτερη απόδοση τάχθηκαν κατά αυτού του είδους της ενθυλάκωσης, για τον λόγο ότι αυτές οι μέθοδοι είχα πολύ χαμηλή επίδοση. Ευτυχώς, τώρα οι εικονικές μηχανές της Java πραγματοποιούν συμπύκνωση του κώδικα σε τέτοιου είδους μεθόδους. Ως εκ τούτου μπορεί να γράψει κάνεις τον παρακάτω κώδικα:

4.2.3 Μεταγλωττιστής HotSpot[7]

Το όνομα HotSpot προέρχεται από την προσέγγιση που γίνεται για να μεταγλωττιστεί ο κώδικας. Σε ένα συνηθισμένο πρόγραμμα, μόνο ένα μικρό κομμάτι κώδικα εκτελείται πολύ συχνά, και η επίδοση στην ταχύτητα εξαρτάται κυρίως από το πόσο γρήγορα εκτελούνται αυτά τα κομμάτια κώδικα. Αυτά τα κρίσιμα τμήματα είναι γνωστά ως τα καυτά σημεία της εφαρμογής, όσο πιο συχνά εκτελείται ένα τμήμα κώδικα, τόσο πιο καυτό θεωρείται.

Ως εκ τούτου, όταν το εικονικό μηχάνημα της Java εκτελέσει τον κώδικα, δεν ξεκινάει την μεταγλώττιση αμέσως. Υπάρχουν δύο βασικοί λόγοι για αυτό. Πρώτον, αν ο κώδικας εκτελεστεί μόνο μία φορά, τότε η μεταγλώττιση είναι ανούσια. Θα είναι πολύ πιο γρήγορο να ερμηνευτεί ο Java bytecode κατευθείαν. Αν όμως ο κώδικας είναι μία μέθοδος που καλείται συνέχεια, ή μία δομή επανάληψης που εκτελείται πολλές φορές, τότε η μεταγλώττιση αξίζει, οι κύκλοι μηχανής που θα χρειαστούν για να γίνει αυτό υπερκαλύπτουν τον χρόνο που θα χανόταν κατά την διάρκεια των επαναλήψεων ενός μη μεταγλωττισμένου κώδικα. Αυτός είναι και ο κύριος λόγος ύπαρξης του μεταγλωττιστή, να μπορεί να διακρίνει τα σημεία ή τις μεθόδους που καλούνται συχνά ώστε να βελτιστοποιήσει την εκτέλεσή τους. Αρχικά κοιτάει τον κώδικα ολόκληρο, έτσι ξέρει κατά την διάρκεια της μεταγλώττισης που γίνεται ύστερα, ποια είναι τα καυτά σημεία.

Ο δεύτερος βασικός λόγος είναι ο εξής, όσες περισσότερες φορές εκτελείται μία μέθοδος ή μία δομή επανάληψης από την εικονική μηχανή, τόσες περισσότερες πληροφορίες έχει για τον κώδικα. Αυτό επιτρέπει στην εικονική μηχανή να κάνει έναν αισθητό αριθμό βελτιστοποιήσεων πάνω στον κώδικα που θα παράγει μετά την μεταγλώττιση. Ένα απλό παράδειγμα είναι το εξής, ας εξετάσουμε την περίπτωση της μεθόδου equals(), η οποία ελέγχει αν δύο αντικείμενα είναι ίδια*[4]*. Αυτή η μέθοδος υπάρχει σε όλα τα αντικείμενα της Java (κληρονομείται από την κλάση Object) και συχνά παρακάμπτεται. Όταν ο διερμηνέας συναντήσει την εντολή b= obj1.equals(obj2); πρέπει να είναι γνωστός ο τύπος του αντικειμένου obj1 προκειμένου να εκτελεστεί η

ανάλογη μέθοδος. Αυτή η δυναμική αναζήτηση είναι κάπως χρονοβόρα. Με τους τρόπους που αναφέρθηκαν προηγουμένως, το εικονικό μηχάνημα μπορεί να γνωρίζει ότι το obj1 είναι τύπου java.lang.String. Έτσι το εικονικό μηχάνημα θα παράγει κώδικα ο οποίος θα καλέσει κατευθείαν την μέθοδο String.equals(). Τώρα ο κώδικας είναι πολύ πιο γρήγορος, όχι μόνο επειδή έχει γίνει μεταγλώττιση, αλλά επειδή μπορεί να παρακάμψει την αναζήτηση της σωστής μεθόδου.

Δεν είναι τόσο απλό βέβαια, είναι πιθανό την επόμενη φορά που ο κώδικας θα εκτελεστεί το obj1 να μην είναι String και για αυτόν τον λόγο το εικονικό μηχάνημα να πρέπει να παράγει μεταγλωττισμένο κώδικα ο οποίος θα εξετάζει αυτήν την πιθανότητα. Όπως και να έχει, ο τελικός κώδικας μετά την μεταγλώττιση θα είναι πιο γρήγορος, γιατί αποφεύγεται η αναζήτηση του τύπου του αντικειμένου, αυτού του τύπου βελτιστοποίηση μπορεί να γίνει μόνο αφού γίνει μία πρώτη σάρωση του κώδικα.

Καταχωρητές και κύρια μνήμη

Μία από τις πιο σημαντικές βελτιστοποιήσεις που κάνει ο μεταγλωττιστής είναι δράσεις που έχουν να κάνουν με την ανάκτηση και εγγραφή τιμών από την μνήμη. Θεωρείστε τον παρακάτω κώδικα:

```
public class RegisterTest
{
    private int sum;

    Public void calculateSum(int n)
    {
        for (int i = 0; i < n; i++)
        {
            Sum += 1 ;
        }
    }
}
```

Σε κάποια χρονική στιγμή, η μεταβλητή sum πρέπει να καταχωρηθεί στην μνήμη, αλλά η ανάκτηση της τιμής είναι ακριβή λειτουργία και χρειάζεται αρκετούς κύκλους μηχανής για να ολοκληρωθεί. Εάν η τιμή του sum έπρεπε να ανακτηθεί και ύστερα να εγγραφεί στην κύρια μνήμη σε κάθε επανάληψη, η απόδοση του προγράμματος θα ήταν δραματικά αργή. Αντ' αυτού, ο μεταγλωττιστής θα φορτώσει έναν καταχωρητή με την αρχική

τιμή του sum, θα πραγματοποιήσει τις επαναλήψεις χρησιμοποιώντας τις τιμές του καταχωρητή, και μετά θα αποθηκεύσει το αποτέλεσμα από τον καταχωρητή στην κύρια μνήμη.

Αυτού του είδους βελτιστοποίηση είναι πολύ αποτελεσματική, αλλά αυτό σημαίνει ότι πρέπει να ληφθεί υπόψιν ο συγχρονισμός τον νημάτων (threads). Ένα νήμα δεν μπορεί να δει την τιμή της μεταβλητής όσο είναι αποθηκευμένη στον καταχωρητή, εφόσον ανήκει σε άλλο νήμα. Ο συγχρονισμός καθιστά εφικτό την δήλωση ότι η μεταβλητή sum είναι διαθέσιμη στην κύρια μνήμη για τα υπόλοιπα νήματα.

4.2.4 Τεχνικές βελτιστοποίησης εικονικών μηχανών[8]

Πολλές βελτιστοποιήσεις έχουν αυξάνει την απόδοση της Java εικονικής μηχανής (JVM) με την πάροδο του χρόνου. Ωστόσο, παρόλο που η Java ήταν η πρώτη που υλοποίησε την εικονική μηχανή με επιτυχία, έχουν χρησιμοποιηθεί συχνά και σε άλλες παρόμοιες πλατφόρμες.

Βελτιστοποίηση με Just-in-time comilation

Πρώιμες Java εικονικές μηχανές πάντα ερμήνευαν bytecodes. Αυτό είχε μία ποινή στην απόδοση δέκα με είκοσι φορές στην Java σε σχέση με την C κατά μέσο όρο στις εφαρμογές. Για να αντιμετωπιστεί αυτό, δημιουργήθηκε ο Just-In-Time (JIT) μεταγλωττιστής στην Java 1.1. Λόγω του υψηλού κόστους της μεταγλώττισης, ένα επιπλέον σύστημα εν ονόματι HotSpot εισήχθη στην Java 1.2 το οποίο στην επόμενη έκδοση -Java 1.3- έγινε προεπιλεγμένο. Χρησιμοποιώντας αυτό το πλαίσιο (framework), η εικονική μηχανή αναλύει την απόδοση του προγράμματος για "καυτά σημεία" (hot spots) που εκτελούνται συχνά ή επανειλημμένα. Στην συνέχεια αυτά τα σημεία βελτιστοποιούνται, κάτι το οποίο οδηγεί σε μία εκτέλεση υψηλής απόδοσης με ελάχιστη επιβάρυνση σε κρίσιμο κώδικας χαμηλής απόδοσης. Κάποια εργαλεία που αξιολογούν την απόδοση δείχνουν ότι αυτή η τεχνική αποφέρει μία αύξηση της ταχύτητας κατά δέκα φορές. Ωστόσο, λόγω των χρονικών περιορισμών, ο μεταγλωττιστής δεν μπορεί να βελτιστοποιήσει πλήρως το πρόγραμμα, ως εκ τούτου το πρόγραμμα που προκύπτει είναι πιο αργό από τον μητρικό κώδικα του εκάστοτε επεξεργαστή.

Προσαρμοσμένη βελτιστοποίηση[9]

Η προσαρμοσμένη βελτιστοποίηση είναι μία τεχνική στην επιστήμη των ηλεκτρονικών υπολογιστών που πραγματοποιεί δυναμική επαναμεταγλώττιση τμημάτων του προγράμματος με βάση το παρόν εκτελέσιμο προφίλ. Με μία απλή εφαρμογή, η προσαρμοσμένη βελτιστοποίηση μπορεί να κάνει μία απλή ανταλλαγή μεταξύ Just-in-time μεταγλώττισης ή να ερμηνεύσει τις οδηγίες. Σε ένα άλλο επίπεδο, η προσαρμοσμένη βελτιστοποίηση μπορεί να επωφεληθεί από τις τοπικές συνθήκες των δεδομένων ώστε να βελτιστοποιήσει τις διακλαδώσεις και να χρησιμοποιήσει επεκτάσεις.

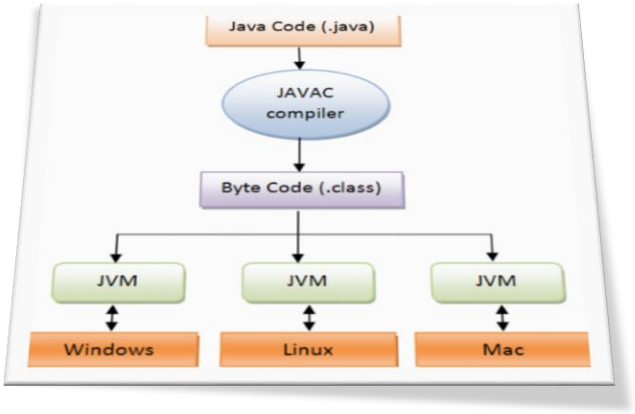
Μία εικονική μηχανή όπως το HotSpot έχει την δυνατότητα να απο-βελτιστοποιήσει τον κώδικα που έγινε μεταγλώττιση JIT προηγουμένως. Αυτό επιτρέπει να γίνει επιθετική (και ενδεχομένως μή ασφαλή) βελτιστοποίηση, ενώ εξακολουθεί να έχει την δυνατότητα να απο-βελτιστοποιήσει τον κώδικα και να επιστρέψει σε κάποιο ασφαλές μονοπάτι αργότερα.

Βελτιστοποίηση συλλογής απορριμμάτων (The Garbage collection)

Οι εικονικές μηχανές 1.0 και 1.1 χρησιμοποιούσαν έναν συλλέκτη σκουπιδιών εν ονόματι mark-sweep, ο οποίος μπορούσε να κατακερματίσει τον σωρό μετά απο ένα ξερκαρτάρισμα. Με την Java 1.2, οι εικονικές μηχανές άλλαξαν σε έναν μεταγεννέστερο συλλέκτη, ο οποίος έχει πολύ καλήτερη συμπεριφορά στην ανασυγκρότιση. Οι μοντέρνες εικονικές μηχανές χρησιμοποιούν μία ποικιλία τεχνικών που βελτιώνουν την απόδοση της συλλογής απορριμάτων.

Άλλες τεχνικές βελτιστοποίησης

- **Compressed Oops**
- **Split bytecode verification**
- **Escape analysis and lock coarsening**
- **Register allocation improvements**
- **Class data sharing**



Εικόνα 3 - Αρχιτεκτονική Java

4.2.5 Σύγκριση με άλλες γλώσσες[10]

Αντικειμενικά, το να συγκρίνεις την επίδοση ενώς Java προγράμματος με ένα άλλο ισοδύναμο το οποίο είναι γραμμένο σε άλλη γλώσσα, όπως την C++, απαιτεί ένα καλά κατασκευασμένο εργαλείο αξιολόγισης της απόδοσης το οποίο συγκρίνει με κάποιον τρόπο τους προγραμματιστικούς αλγόριθμους σε τεχνικό επίπεδο. Ο στόχος του bytecode μεταγλωττιστή της Java είναι η ίδια η πλατφόρμα της Java, και ο κώδικας bytecode είτε ερμηνεύεται είτε καταρτίζεται σε κώδικα μηχανής από το JVM. Άλλοι μεταγλωττιστές σχεδόν πάντα έχουν στόχο ένα συγκεκριμένο υλικό και λογισμικό, δημιουργώντας έτσι κώδικα σε γλώσσα μηχανής που θα μείνει αμετάβλητος κατά την διάρκεια της εκτέλεσης. Προκύπτουν έτσι πολλά διαφορετικά σενάρια, δύσκολα στην σύγκριση: στατικές εναντίον δυναμικές μεταγλωττίσεις και επαναμεταγλωττίσεις, διαθεσιμότητα ακριβής πληροφορίας για το περιβάλλον που εκτελείται ένα πρόγραμμα και άλλα.

Η Java μεταγλωττίζεται κατά την διάρκεια της εκτέλεσης από την εικονική μηχανή αυτής, αλλά μπορεί σε άλλες περιπτώσεις να μεταγλωττιστεί μεταγενέστερα, όπως συμβαίνει στην C++. Όταν η μεταγλώττιση γίνεται εκείνη την στιγμή η επίδοση της Java σε σχέση με άλλες γλώσσες καταγράφεται ως εξής:

- **Πιό αργή ή παρόμια επίδοση απο παρόμοιες γλώσσες όπως η C ή η C++,**
- **Παρόμοια επίδοση σε σχέση με άλλες γλώσσες που κάνουν μεταγλώττιση εκείνη την στιγμή όπως η C#,**
- **Πολύ πιο γρήγορη επίδοση απο γλώσσες που δεν έχουν αποτελεσματική μεταγλώττιση σε εγγενή κώδικα, όπως η Perl, η Ruby, η PHP και η Python.**

Ταχύτητα προγράμματος

Εργαλεία αξιολόγησης συχνά μετρούν την επίδοση σε μικρά προγράμματα που τρέχουν βαριές μαθηματικές πράξεις. Σε μερικές περιπτώσεις σε πραγματικά προγράμματα, η Java έχει καλύτερη επίδοση από την C. Ένα παράδειγμα είναι το εργαλείο αξιολόγησης ονόματι Jake2, ένας κλώνος του Quake 2 γραμμένος σε Java, που έχει προέλθει από την μεταγλώττιση του κώδικα GPL C. Η 5.0 έκδοση της Java αποδίδει καλύτερα σε κάποια μηχανήματα υπολογιστών σε σχέση με την C. Δεν είναι ακριβές κάτω από ποιες συνθήκες έγιναν αυτές οι μετρήσεις (για παράδειγμα αν χρησιμοποιήθηκε το πρωτότυπο Quake 2 που γράφτηκε το 1997, το οποίο ίσως να είναι κακό δείγμα αφού μεταγενέστεροι μεταγλωττιστές της C έχουν βελτιστοποιήσεις για το Quake), φαίνεται όμως ότι ο ίδιος πηγαίος κώδικας σε Java μπορεί να έχει μεγάλη ώθηση στην ταχύτητα απλά αναβαθμίζοντας το εικονικό μηχάνημα, κάτι το οποίο είναι αδύνατο με μία στατική προσέγγιση.

Σε άλλα προγράμματα ένα C++ μπορεί και συνήθως είναι πολύ πιο γρήγορο από το ισοδύναμο Java πρόγραμμα. Μία σύγκριση που έγινε από την Google το 2011 έδειξε ότι η C++ είναι 10 φορές πιο γρήγορη από την Java. Σε αντιπαράθεση με μία ακαδημαϊκή σύγκριση που έγινε το 2012 πάνω σε έναν αλγόριθμο για τρισδιάστατα γραφικά η οποία έδειξε ότι στα Windows το εικονικό μηχάνημα Java 6 είναι 1,09 με 1,91 φορές πιο αργό από την C++.

Κάποιες βελτιστοποιήσεις οι οποίες είναι πιθανές στην Java και σε παρόμοιες γλώσσες μπορεί να μην είναι δυνατές κάτω από ορισμένες προϋποθέσεις στην C++:

- **Χρήση του δείκτη C-style εμποδίζει την βελτιστοποίηση σε γλώσσες που υποστηρίζουν τους δείκτες,**
- **Χρήση των τεχνικών ανάλυσης διαφυγής είναι περιορισμένη στην C++, επειδή για παράδειγμα ο μεταγλωττιστής C++ δεν μπορεί πάντα να γνωρίζει άν ένα αντικείμενο θα αλλαχτεί σε κάποιο κομμάτι κώδικα λόγω των δεικτών,**
- **Η Java έχει πρόσβαση σε συμπληρωματικά στιγμιότυπα μεθόδων πιο γρήγορα από την C++. Ωστόσο, για μη εικονικές μεθόδους η C++ έχει καλύτερη επίδοση από την Java.**

Τα εικονικά μηχανήματα της Java επίσης εκτελούν επεξεργασίες και βελτιστοποιήσεις για τον συγκεκριμένο επεξεργαστή. Αυτή η ικανότητα δίνει την ικανότητα στην Java να κάνει πιο επιθετικές και δραστικές βελτιστοποιήσεις σε σχέση με αυτές που θα γίνουν από μία γλώσσα στατικού τύπου, ειδικά όταν θα αναμειχτούν εξωτερικές βιβλιοθήκες.

Τα αποτελέσματα αξιολόγησης μεταξύ την Java και της C++ εξαρτώνται από τί διαδικασίες συγκρίνονται. Για παράδειγμα, όταν γίνει σύγκριση με την Java 5.0

- **Σε λειτουργικά συστήματα 32 bit και 64 bit οι αριθμητικές πράξεις, η εισαγωγή/εξαγωγή σε αρχεία και ο χειρισμός των Exceptions, έχουν παρόμοια επίδοση σε σχέση με ένα πρόγραμμα C++,**
- **Ο χειρισμός των πινάκων είναι καλύτερος στην C,**
- **Τριγονομεντρικές πράξεις επίσης είναι πολύ καλύτερες στην C.**

Επιδόσεις σε υπολογιστές με πολλούς πυρήνες

Η επεκτασιμότητα και η επίδοση των Java εφαρμογών σε συστήματα με πολλούς πυρήνες περιορίζεται από τον ρυθμό κατανομής των αντικειμένων(Objects). Αυτό το φαινόμενο ονομάζεται "τοίχος κατανομής" (Allocation wall). Ωστόσο στην πράξη, οι σύγχρονοι αλγόριθμοι συλλογής απορριμμάτων χρησιμοποιούν πολλούς πυρήνες για να πραγματοποιήσουν τον σκοπό τους. Κάποιοι από αυτούς, έχει αναφερθεί ότι κρατούν τον ρυθμό κατανομής σε πολύ ψηλά επίπεδα, πάνω από ένα gigabyte το δευτερόλεπτο, καθώς επίσης υπάρχουν και συστήματα που βασίζονται σε Java τα οποία δεν έχουν πρόβλημα να χρησιμοποιήσουν μερικές εκατοντάδες πυρήνες.

Η αυτόματη διαχείριση της μνήμης στην Java, επιτρέπει την αποτελεσματική χρήση των αμετάβλητων δομών δεδομένων που συχνά είναι απίθανο να πραγματοποιθεί εκκαθάριση σκουπιδιών πάνω σε αυτές. Η Java προσφέρει ένα μεγάλο πλήθος δομών υψηλού επιπέδου στην βασική της βιβλιοθήκη στο πακέτο java.util.concurrent, ενώ άλλες γλώσσες που χρησιμοποιούνται για απόδοση όπως η C ή η C++ ακόμα στερούνται κάτι τέτοιο.

Χρόνος εκκίνησης

Ο χρόνος εκκίνησης της Java είναι συχνά πολύ περισσότερος από πολλές γλώσσες, συμπεριλαμβανομένου την C, την C++, την Perl ή την Python, επειδή πρίν να χρησιμοποιηθούν πρέπει να φορτωθούν πολλές κλάσεις και πρώτα απ' όλα κλάσεις από την βιβλιοθήκη της πλατφόρμας.

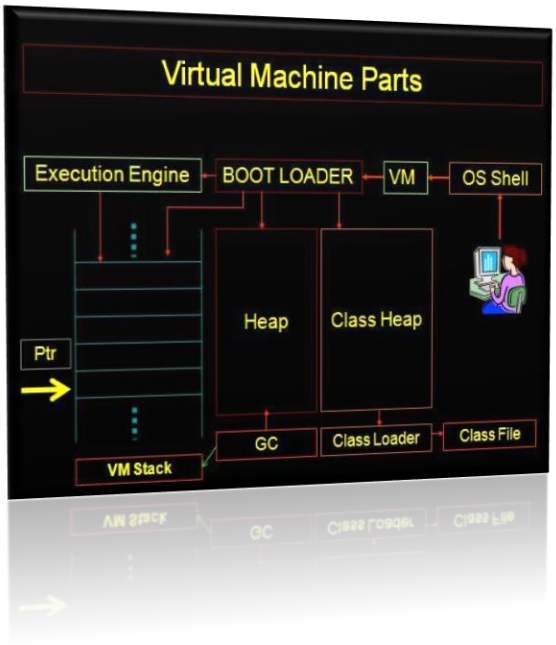
Σε σύγκριση με παρόμοιους χρόνους εκτέλεσης, για μικρά προγράμματα σε ένα μηχάνημα Windows, ο χρόνος εκκίνησης είναι περίπου ο ίδιος με το Mono και λίγο πιο αργό από το Net.

Φαίνεται ότι ο περισσότερος χρόνος εκκίνησης οφείλεται κατά ένα μεγάλο ποσοστό σε διαδικασίες εισόδου-εξόδου παρά στην αρχικοποίηση της εικονικής μηχανής της Java ή το φόρτωμα των κλάσεων (το αρχείο δεδομένων rt.jar από μόνο του είναι 40 MB και η εικονική μηχανή πρέπει να αναζητήσει πολλά δεδομένα από αυτό το τεράστιο αρχείο). Μερικές δοκιμές έδειξαν πως παρά την υλοποίηση της καινούριας τεχνικής Split bytecde verification, η φόρτωση βελτιώθηκε το πολύ κατά 40%, κάτι το οποίο μεταφράζεται σε περίπου 5% βελτίωση στην εκκίνηση για μεγάλα προγράμματα. Αν και μία μικρή βελτίωση είναι πιο αισθητή σε μικρά προγράμματα τα οποία εκτελούν μία μικρή διαδικασία και μετά τερματίζονται.

Ξεκινώντας απο την Java SE 6 έκδοση 10, το περιβάλλον εκτέλεσης (JRE) της Sun περιέχει κλάσεις δεδομένων στην εκκίνηση του λειτουργικού συστήματος, ώστε τα δεδομένα να βρίσκονται στην μνήμη και όχι στον δίσκο. Με αυτόν τον τρόπο η ανάκτηση των δεδομένων αυτών έχει πολύ μικρότερο χρονικό κόστος.

Η επικρατέστερη προσέγγιση JET αντιμετωπίζει το πρόβλημα από άλλη οπτική γωνία. Ένα σύστημα βελτιστοποίησης μειώνει το πλήθος των δεδομένων που πρέπει να διαβαστούν από τον δίσκο κατά την διάρκεια της εκκίνησης, και κάνει τις αναζητήσεις ακολουθία.

Τον Νοέμβριο του 2004, κυκλοφόρησε στο ευρύ κοινό το Nailgun , ένα client-server πρωτόκολλο το οποίο τρέχει τα προγράμματα Java από την γραμμή εντολών χωρίς να εμπεριέχει το καπέλο της εκκίνησης της εικονικής μηχανής της Java. Εισήχθη έτσι για πρώτη φορά η δυνατότητα σε ένα Script να χρησιμοποιήσει την εικονική μηχανή της Java σαν το Daemon. Το Nailgun daemon δεν είναι ασφαλές, όλα τα προγράμματα τρέχουν με τα ίδια δικαιώματα που έχει ένας Server, για αυτό με την υιοθέτηση του Nailgun είναι απαραίτητες περεταίρω προφυλάξεις. Script τα οποία η εκκίνηση των εικονικών μηχανών ανά εφαρμογή γονατίζει τους διαθέσιμου πόρους του υλικού μηχανήματος, έχουν βελτίωση στον χρόνο εκτέλεσης μία με δύο φορές.



Εικόνα 4 - Κομμάτια εικονικών μηχανών

Χρήση της μνήμης

Η χρήση της μνήμης στην Java είναι πολύ πιο βαριά απο αυτή της C++ επειδή:

- **Υπάρχει ένα καπέλο 8 bytes για κάθε αντικείμενο και 12 bytes για κάθε πίνακα στην Java. Αν το μέγεθος του αντικειμένου δεν είναι πολλαπλάσιο των 8 bytes, στρογγυλοποιείται προς τα πάνω στο επόμενο πολλαπλάσιο του 8. Αυτό σημαίνει ότι ένα αντικείμενο που περιέχει ένα byte, καταλαμβάνει 16 bytes και απαιτεί 4 bytes για τον δείκτη αναφοράς. Σημειωτέων ότι και η C++**

χρησιμοποιεί δείκτες που συνήθως καταλαμβάνουν 4 με 8 bytes, για κάθε αντικείμενο που δηλώνει εικονικές λειτουργίες.

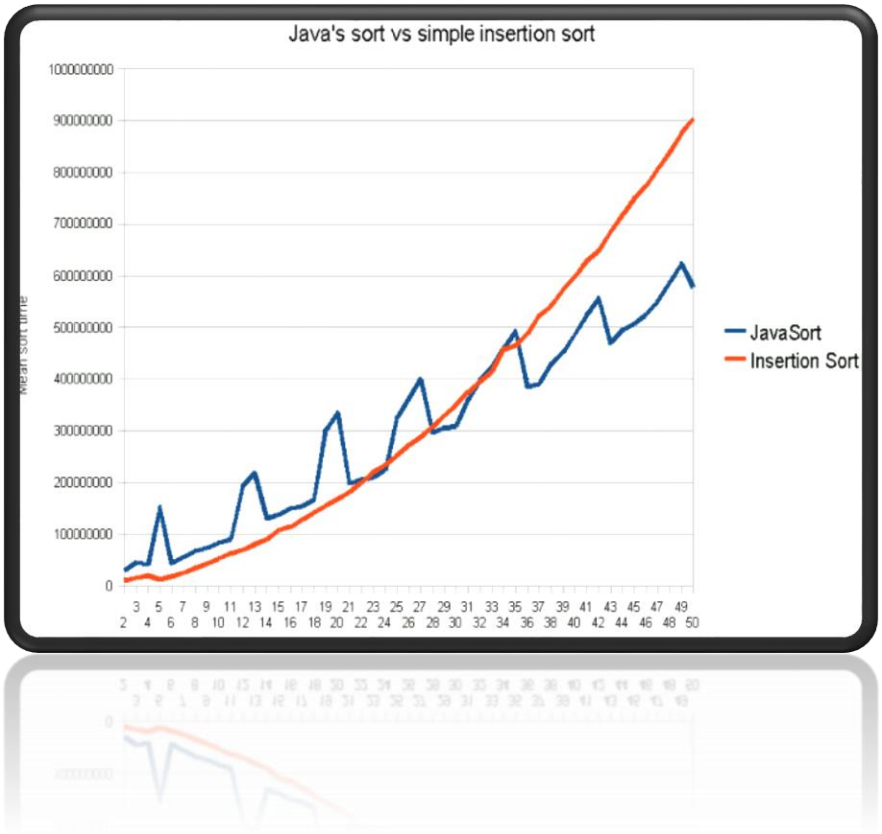
- Κομμάτια από τις βιβλιοθήκες της Java πρέπει να φορτωθούν πριν την εκτέλεση του προγράμματος, τουλάχιστον κλάσεις που χρησιμοποιούνται στο φόντο από το πρόγραμμα. Αυτό δημιουργεί ένα μεγάλο καπέλο στις μικρές εφαρμογές.
- Τόσο το binary της Java όσο και ο εγγενής κώδικας που προέρχεται απο την μεταγλώττιση βρίσκεται τυπικά στην μνήμη.
- Η ίδια η εικονική μηχανή καταναλώνει ένα σημαντικό χώρο στην μνήμη.
- Στην Java, ένα σύνθετο αντικείμενο (μία κλάση A που χρησιμοποιεί στιγμιότυπα του Β και του Γ) δημιουργείται χρησιμοποιώντας αναφορές κατανεμημένων στιγμιότυπων του Β και του Γ. Στην C++ η μνήμη και το κόστος απόδοσης αυτών των αναφορών μπορεί να αποφευχθεί με την ύπαρξη του στιγμιότυπου Β και Γ μέσα στο Α.
- Η έλλειψη αριθμητικών διευθύνσεων καθιστά απίθανη την δημιουργία δομών που να είναι αποδοτικές για την μνήμη· βλέπε λίστες που συνδέονται με XOR.

Στις περισσότερες περιπτώσεις μία εφαρμογή σε C++ θα καταναλώσει πολύ λιγότερη μνήμη από την αντίστοιχη σε Java λόγω της ύπαρξης της εικονικής μηχανής, το φόρτωμα των κλάσεων και την αυτόματη αυξομείωση της μνήμης. Σε εφαρμογές που η μνήμη είναι κρίσιμος παράγοντας, για την επιλογή της γλώσσας στο περιβάλλον εκτέλεσης απαιτείται ανάλυση κόστους σε σχέση με το όφελος.

Η χρησιμότητα σε υπολογισμούς υψηλών απαιτήσεων

Μερικοί πιστεύουν ότι η επίδοση της Java σε υπολογιστές υψηλής απόδοσης είναι παρόμοια με την Fortran για βαριούς υπολογισμούς, αλλά ότι τα εικονικά μηχανήματα της Java ακόμα έχουν θέματα επεκτασιμότητας σε εκτενείς διαδικασίες επικοινωνίας σε δίκτυα πλέγματος.

Ωστόσο, εφαρμογές που κάνουν υπολογισμούς υψηλής απόδοσης και είναι γραμμένες σε Java, πρόσφατα κέρδισαν διαγωνισμούς επιδόσεων. Το 2008 και το 2009, ένας Apache Hadoop (ένα έργο ανοιχτού λογισμικού υψηλών απαιτήσεων γραμμένο σε Java) μπόρεσε να ταξινομήσει terabytes και petabytes ακεραίων το γρηγορότερο δυνατόν. Παρόλο που το υλικό των συστημάτων που ανταγωνίστηκαν δεν ήταν καθορισμένα για αυτόν τον σκοπό.



[6]Εικόνα 5 - Διάγραμμα επιδόσεων Java πάνω σε αλγόριθμο ταξινόμησης

4.2.6 Αυτόματη διαχείριση μνήμης

Η Java χρησιμοποιεί έναν αυτόματο συλλέκτη σκουπιδιών ώστε να διαχειριστεί την μνήμη στον κύκλο ζωής των αντικειμένων. Ο προγραμματιστής καθορίζει πότε τα αντικείμενα θα δημιουργηθούν και η Java σε χρόνο εκτέλεσης είναι υπεύθυνη να ανακτήσει την μνήμη που έχει διαθέσει για αυτά όταν πλέον δεν θα ξανά χρησιμοποιηθούν. Όταν δεν απομένει καμία αναφορά σε ένα αντικείμενο, αυτή η απρόσιτη μνήμη γίνεται διαθέσιμη για να ελευθερωθεί από τον συλλέκτη σκουπιδιών. Κάτι παρόμοιο με διαρροή μνήμης ενδέχεται να γίνει αν ο κώδικας του προγραμματιστή κρατήσει την αναφορά του αντικειμένου παρόλο που δεν χρειάζεται. Εάν κληθεί μία μέθοδος ενός αντικειμένου που δεν είναι διαθέσιμο, τότε ρίπτεται εξαίρεση μηδενικού δείκτη (null pointer exception).

Μία από τις ιδέες πίσω από το μοντέλο της αυτόματης διαχείρισης στην Java, είναι να δώσει στον προγραμματιστή την ευκολία να μην ασχολείται ο ίδιος χειροκίνητα με την διαχείριση της μνήμης. Σε αρκετές γλώσσες, η μνήμη για την δημιουργία αντικειμένων

τοποθετείται στην στοίβα, ή αποκλειστικά στον σωρό. Στην δεύτερη περίπτωση η ευθύνη της διαχείρισης της μνήμης είναι στον προγραμματιστή. Εάν ο προγραμματιστής δεν καθαρίσει ένα αντικείμενο, μπορεί να γίνει διαρροή μνήμης (memory leak). Αν ο προγραμματιστείς προσπαθήσει να αποκτήσει πρόσβαση ή έστω να καθαρίσει ένα αντικείμενο που δεν υπάρχει, τότε το αποτέλεσμα είναι δύσκολο να προβλεφθεί, το πιο πιθανό είναι το πρόγραμμα να γίνει ασταθές και να συνθλίφθεί (crush). Αυτό μπορεί εν μέρει να αντιμετωπιστεί με την χρήση έξυπνων δεικτών, αλλά κάτι τέτοιο προσθέτει εργασία και πολυπλοκότητα. Σημειωτέων, ο συλλέκτης σκουπιδιών δεν αποτρέπει διαρροές μνήμης από λογικά λάθη, για παράδειγμα δεν προβλέπει την εκκαθάριση της μνήμης που παραμένει αναφερόμενη αλλά δεν χρησιμοποιείται ποτέ.

Η εκκαθάριση της μνήμης που δεν χρησιμοποιείται μπορεί να γίνει οποιαδήποτε στιγμή, ιδανικά θα γίνει όταν το πρόγραμμα είναι ανενεργό. Είναι εγγυημένο ότι θα ενεργοποιηθεί όταν δεν υπάρχει ελεύθερη μνήμη στον σωρό για να δημιουργηθεί καινούριο αντικείμενο, κάτι το οποίο μπορεί να καθυστερήσει το πρόγραμμα την στιγμή εκείνη. Ρητή διαχείριση της μνήμης δεν δύναται στην Java.

Η Java δεν υποστηρίζει αριθμητικούς δείκτες όπως η C και η C++, στις οποίες διευθύνσεις για τα αντικείμενα και απόλυτοι ακέραιοι αριθμοί (συνήθως μεγάλοι ακέραιοι) μπορούν να χρησιμοποιηθούν εναλλακτικά. Αυτό επιτρέπει στον συλλέκτη σκουπιδιών να επανατοποθετήσει τα αντικείμενα και έτσι διασφαλίζει τον τύπο αυτών, καθώς επίσης και την ασφάλεια.

Όπως στην C++ και σε άλλες αντικειμενοστραφείς γλώσσες, οι αρχέγονες μεταβλητές (primitive data type) της Java δεν είναι αντικείμενα. Οι τιμές των αρχέγονων τύπων δεδομένων αποθηκεύονται απευθείας σε πεδία (για τα αντικείμενα) ή στην στοίβα (για τις μεθόδους) και όχι στον σωρό. Αυτή ήταν συνειδητή απόφαση των σχεδιαστών της Java για λόγους επίδοσης. Για αυτόν τον λόγο, η Java δεν θεωρείτο αγνή αντικειμενοστραφή γλώσσα προγραμματισμού. Ωστόσο, μετά την Java 5.0 έχει δοθεί η δυνατότητα στους προγραμματιστές να επεξεργάζονται τους αρχέγονους τύπους δεδομένων σαν να είναι στιγμιότυπα μίας κλάσης που τους περιτυλίγει.

Η Java περιλαμβάνει πολλούς τύπους από συλλέκτες σκουπιδιών. Ο προεπιλεγμένος είναι ο HotSpot, ο οποίος χρησιμοποιεί παράλληλες διαδικασίες για να εκπληρώσει τον σκοπό του. Ωστόσο, υπάρχουν και άλλοι τύποι που μπορούν να χρησιμοποιηθούν ώστε να διαχειριστεί ο σωρός. Για το 90% των εφαρμογών σε Java ο συλλέκτης σκουπιδιών Concurrent Mark-Sweep είναι επαρκής. Η Oracle έχει στόχο να αντικαταστήσει τον CMS με τον συλλέκτη Garbage-first (G1).

4.2.7 Ιστορική αναδρομή της Java

Στις αρχές του 1991, η *Sun* αναζητούσε το κατάλληλο εργαλείο για να αποτελέσει την πλατφόρμα ανάπτυξης λογισμικού σε μικρο-συσκευές (έξυπνες οικιακές συσκευές έως πολύπλοκα συστήματα παραγωγής γραφικών). Τα εργαλεία της εποχής ήταν γλώσσες όπως η *C++* και η *C*. Μετά από διάφορους πειραματισμούς προέκυψε το συμπέρασμα ότι οι υπάρχουσες γλώσσες δεν μπορούσαν να καλύψουν τις ανάγκες τους. Ο "πατέρας" της Java, James Gosling, που εργαζόταν εκείνη την εποχή για την Sun, έκανε ήδη πειραματισμούς πάνω στη *C++* και είχε παρουσιάσει κατά καιρούς κάποιες πειραματικές γλώσσες (*C++ ++*) ως πρότυπα για το νέο εργαλείο που αναζητούσαν στην *Sun*. Τελικά μετά από λίγο καιρό κατέληξαν με μια πρόταση για το επιτελείο της εταιρίας, η οποία ήταν η γλώσσα *Oak*. Το όνομά της το πήρε από το ομώνυμο δένδρο (βελανιδιά) το οποίο ο Gosling είχε έξω από το γραφείο του και έβλεπε κάθε μέρα.

Επίσημο σήμα - Oracle

Από την Oak στην Java

Η *Oak* ήταν μία γλώσσα που διατηρούσε μεγάλη συγγένεια με την *C++*. Παρόλα αυτά είχε πολύ πιο έντονο αντικειμενοστραφή (*object oriented*) χαρακτήρα σε σχέση με την *C++* και χαρακτηριζόταν για την απλότητα της. Σύντομα οι υπεύθυνοι ανάπτυξης της νέας γλώσσας ανακάλυψαν ότι το όνομα *Oak* ήταν ήδη κατοχυρωμένο οπότε κατά την διάρκεια μιας εκ των πολλών συναντήσεων σε κάποιο τοπικό καφέ αποφάσισαν να μετονομάσουν το νέο τους δημιούργημα σε Java που εκτός των άλλων ήταν το όνομα της αγαπημένης ποικιλίας καφέ για τους δημιουργούς της. Η επίσημη εμφάνιση της *Java* αλλά και του *HotJava* (πλοηγός με υποστήριξη *Java*) στη βιομηχανία της πληροφορικής έγινε το Μάρτιο του 1995 όταν η *Sun* την ανακοίνωσε στο συνέδριο *Sun*

World 1995. Ο πρώτος μεταγλωττιστής (*compiler*) της ήταν γραμμένος στη γλώσσα C από τον James Gosling. Το 1994, ο A.Van Hoff ξαναγράφει τον μεταγλωττιστή της γλώσσας σε *Java*, ενώ το Δεκέμβριο του 1995 πρώτες οι IBM, Borland, Mitsubishi Electronics, Sybase και Symantec ανακοινώνουν σχέδια να χρησιμοποιήσουν τη *Java* για την δημιουργία λογισμικού. Από εκεί και πέρα η Java ακολουθεί μία ανοδική πορεία και είναι πλέον μία από τις πιο δημοφιλείς γλώσσες στον χώρο της πληροφορικής. Στις 13 Νοεμβρίου του 2006 η Java έγινε πλέον μια γλώσσα ανοιχτού κώδικα (GPL) όσον αφορά το μεταγλωττιστή (javac) και το πακέτο ανάπτυξης (JDK, Java Development Kit).

Η εξαγορά από την Oracle και το μέλλον της Java

Στις 27 Απριλίου 2010 η εταιρία λογισμικού Oracle Corporation ανακοίνωσε ότι μετά από πολύμηνες συζητήσεις ήρθε σε συμφωνία για την εξαγορά της Sun Microsystems και των τεχνολογιών (πνευματικά δικαιώματα/ πατέντες) που η δεύτερη είχε στην κατοχή της ή δημιουργήσει. Η συγκεκριμένη συμφωνία θεωρείται σημαντική για το μέλλον της Java και του γενικότερου οικοσυστήματος τεχνολογιών γύρω από αυτή μιας και ο έμμεσος έλεγχος της τεχνολογίας και η εξέλιξη της περνάει σε άλλα χέρια.

4.3 JSON[11]

Το JSON, είναι μία τυποποιημένη μορφή που χρησιμοποιεί ένα ευκολοδιάβαστο από τον ανθρώπινο μάτι κείμενο για να μεταδώσει αντικείμενα δεδομένων που αποτελούνται από ζεύγη χαρακτηριστικών με τιμών. Είναι η πρωταρχική και επικρατέστερη δομή δεδομένων που χρησιμοποιείται για την ασύγχρονη επικοινωνία μεταξύ browser και server, κυρίως αντικαθιστώντας το XML – όταν χρησιμοποιείται AJAX. Στο έργο που θα αναφερθεί παρακάτω, θα χρειαστεί να χτιστεί μία ασύγχρονη επικοινωνία μεταξύ μίας κινητής συσκευής και ενός server, ο τρόπος υλοποίησής περνάει μέσα από το JSON.

Παρ 'όλο που η γνήσια πηγή δημιουργίας είναι η γλώσσα JavaScript, το JSON είναι μία δομή δεδομένων ανεξαρτήτου γλώσσας. Κώδικας για την ανάλυση και δημιουργία δεδομένων JSON είναι διαθέσιμος ήδη σε πολλές γλώσσες προγραμματισμού.

Η δομή JSON αρχικά ορίστηκε από τον Douglas Crockford. Αυτήν την στιγμή προσδιορίζεται από δύο ανταγωνιστικά πρότυπα, το RFC 7259 και το ECMA-404. Το ECMA πρότυπο είναι μινιμαλιστικό, περιγράφοντας μόνο την επιτρεπόμενη γραμματική σύνταξη, ενώ το RFC επίσης προσφέρει κάποιες σημασιολογικές έννοιες και εκτιμήσεις ασφάλειας. Ο επίσημος τύπος μέσου διάδοσης για το JSON είναι το application/json. Η επέκταση του ονόματος του αρχείου αυτού είναι το .json.

Παράδειγμα Json

Το παρακάτω παράδειγμα δείχνει μία πιθανή απεικόνιση ενός JSON αντικειμένου, που περιγράφει ένα άτομο.

```
{
        "firstName": "John",
        "lastName": "Smith",
        "isAlive": true,
        "age": 25,
        "address": {
                "streetAddress": "21 2nd Street",
                "city": "New York",
                "state": "NY",
                "postalCode": "10021-3100"
        },
        "phoneNumbers": [ {
                "type": "home",
                "number": "212 555-1234"
        },
        {
                "type": "office",
                "number": "646 555-4567"
        } ],
        "children": [],
        "spouse": null
}
```

4.4 Υλοποίηση Json του προγράμματος Tv vote cast

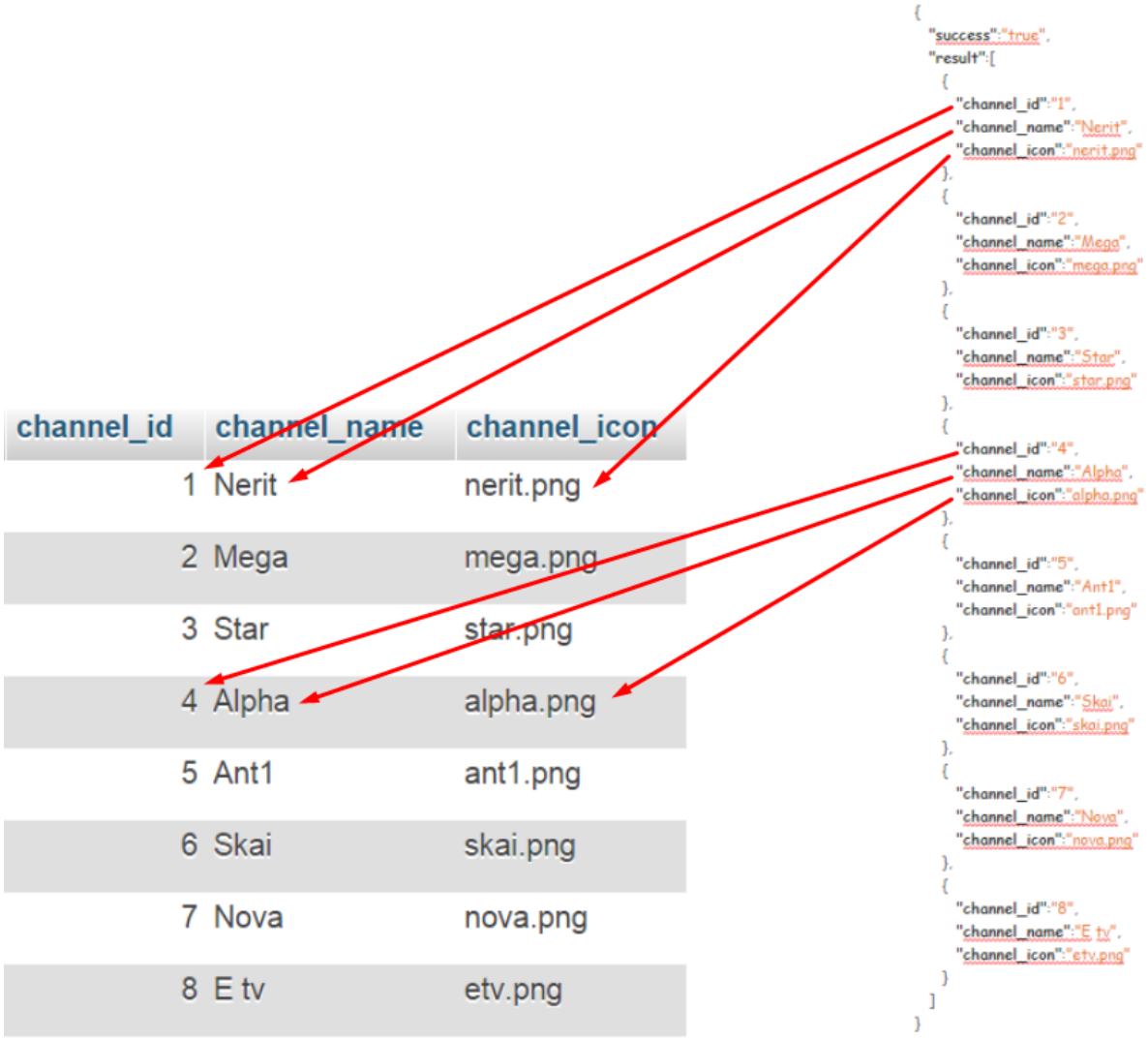
Το Json είναι μία εύκολη τεχνική ώστε να μπορέσουν δύο διαφορετικά συστήματα να επικοινωνήσουν. Ειδικά στον κόσμου του σήμερα, η διαδρομή πληροφορίας ξεκινάει από μία άγνωστη συσκευή – συνήθως έναν Server – και καταλήγει σε μία ή εκατοντάδες άλλες. Όλοι αυτοί οι σταθμοί έχουν διαφορετικό hardware και διαφορετικό software. Για παράδειγμα τα κινητά σήμερα είναι κατασκευασμένα από διαφορετικές εταιρείες, που η κάθε μία αποφασίζει με τους δικούς της κανόνες το πώς θα κατασκευάσει το

προϊόν. Από μεριά software υπάρχουν τα Android, τα iOS, τα Windows και άλλα τα οποία είτε έχουν εκλείψει στην αγορά είτε ακόμα δεν τα γνωρίζουμε τόσο καλά γιατί είναι σε πρώιμο στάδιο. Από γλώσσες προγραμματισμού έχουν επικρατήσει η C, C++, C#, java, Visual basic, php, javascript, xml, css, objective C, oracle sql κ.α. Όλες οι παραπάνω έχουν πολλές διαφορές, κάτι το οποίο δημιουργεί το πρόβλημα της επικοινωνίας μεταξύ δύο εκ των παραπάνω. Στην εποχή της τεχνολογίας, που ο καθένας έχει την δική του ηλεκτρονική συσκευή, όλες αυτές οι γλώσσες πρέπει να βρουν έναν κοινό άξονα ώστε να μπορεί να γίνει ανταλλαγή δεδομένων με κάποιον τρόπο. Ένα κομμάτι της λύσης είναι το Json.

Όλες οι γλώσσες προγραμματισμού έχουν κάτι κοινό, την έννοια του String. Είναι το κείμενο το οποίο βλέπουμε συνεχώς στην έξοδο ενός συστήματος, δηλαδή μία οθόνη, μία ηλεκτρονική πινακίδα, ένα ψηφιακό ρολόι. Το String είναι ένα από τους πρωτόγονους τύπους δεδομένων (primtive data type) και για αυτό υπάρχει σε όλες τις γλώσσες, αλλά ένα σύστημα για να ανταλλάξει πληροφορία χρειάζεται στην φαρέτρα του κάτι ποιο σύνθετο από μία απλή γραμμή κειμένου για να ανταλλάξει μια φωτογραφία για παράδειγμα. Η πληροφορία επίσης απαιτείται να είναι δομημένη και συμφωνημένη γιατί όπως προαναφέρθηκε είναι ο μόνος τρόπος να επικοινωνήσουν δύο "ξένοι". Σε μία πληθώρα των περιπτώσεων η πληροφορία μεταφέρεται μέσα σε δομημένους πίνακες, όπως και αποθηκεύεται σε ένα σκληρό δίσκο ή σε μία ηλεκτρονική μνήμη γενικότερα. Το Json είναι ένα κείμενο που περιγράφει έναν πίνακα. Κάτι πολύ παρόμοιο με xml δηλαδή, πολύ πιο απλό, πιο γρήγορο, όχι τόσο ασφαλές.

Ένα τέτοιο "script" χρησιμοποιεί η εφαρμογή Tv vote cast για να μπορούν δύο διαφορετικά συστήματα να έρθουν σε επικοινωνία. Από την μία μεριά υπάρχει το κινητό το οποίο είναι android, άρα εκτελεί Java κώδικα, από την άλλη μεριά έχουμε τον Server ο οποίος εκτελεί php. Σκοπός του Server είναι να δέχεται αιτήσεις από το κινητό και να δίνει την ανάλογη απάντηση. Όταν για παράδειγμα το μενού του κινητού δείχνει την λίστα των καναλιών και τα εικονίδια τους, θα πρέπει να ρωτήσει τον Server ποια θα είναι αυτά.

Στο σενάριο αυτό που είναι κομμάτι της εφαρμογής, το Json είναι το ακόλουθο (Εικόνα 6). Παρουσιάζεται σε αντιστοιχία με τον αντίστοιχο πίνακα της βάσης δεδομένων.



Εικόνα 6 - Αντιστοίχιση Json & βάση δεδομένων

Το παρακάτω παρουσιάζει το Json αποτέλεσμα του query

```
SELECT * FROM `program_view`;
```

```
{
  "success":true,
  "result":[
    {
      "tv_program_id":"1",
      "channel_id":"1",
      "channel_name":"Nerit",
      "channel_icon":"nerit.png",
      "show_id":"1",
      "show_name":"Nerit News",
      "show_start":"13:00",
      "show_end":"14:00",
      "vote_rating":"3.1346153846153846"
    },
    {
      "tv_program_id":"1",
      "channel_id":"1",
      "channel_name":"Nerit",
      "channel_icon":"nerit.png",
      "show_id":"1",
      "show_name":"Nerit News",
      "show_start":"20:00",
      "show_end":"21:00",
      "vote_rating":"3.1346153846153846"
    }
  ]
}
```


5. SERVER SIDE

Όλα τα παραπάνω αναφέρονται και περιγράφουν τί γίνεται στον χρήστη (client side). δηλαδή την τηλεόραση, τον άνθρωπο, το κινητό. Το υπόλοιπο μισό του συστήματος βρίσκεται σε ένα σταθερό σύστημα, το οποίο θα εξυπηρετεί τους χρήστες, αυτή η μεριά στην γλώσσα της πληροφορικής ονομάζεται server side. Στην σήμερον ημέρα ένα τέτοιο σύστημα αποτελείται από υπολογιστές που τρέχουν έναν Server που μπορεί να επεξεργαστεί πολλά δεδομένα. Ένας από τους πιο γνωστούς που θα χρησιμοποιηθεί για τις ανάγκες του έργου, είναι ο Apache server. Λόγω του ότι δεν υπάρχουν τα

απαραίτητα κονδύλια για την μίσθωση ενός "σοβαρού" server, ένας απλός προσωπικός υπολογιστής θα χρησιμοποιηθεί για να διαδραματίσει αυτόν τον ρόλο.

5.1 Ο ορισμός του Server [12]

Εξυπηρετητής ή διακομιστής (αγγλ.: server) είναι υλικό ή / και λογισμικό που αναλαμβάνει την παροχή διάφορων υπηρεσιών, «εξυπηρετώντας» αιτήσεις άλλων προγραμμάτων, γνωστούς ως πελάτες (clients) που μπορούν να τρέχουν στον ίδιο υπολογιστή ή σε σύνδεση μέσω δικτύου. Όταν ένας υπολογιστής εκτελεί κυρίως τέτοια προγράμματα εξυπηρετητές συνεχόμενα, 24 ώρες την ημέρα, τότε μπορούμε να αναφερθούμε σε όλον τον υπολογιστή ως *εξυπηρετητή*, αφού αυτή είναι η κύρια λειτουργία του. Παρομοίως, ως *πελάτη* μπορούμε να θεωρήσουμε είτε κάποιο λογισμικό που επικοινωνεί και υποβάλει αιτήματα στον εξυπηρετητή, είτε σε όλο τον υπολογιστή όταν ο εξυπηρετητής είναι άλλος υπολογιστής και οι 2 υπολογιστές είναι συνδεδεμένοι σε ένα δίκτυο.

Ο όρος εξυπηρετητής χρησιμοποιείται ευρέως στον χώρο της τεχνολογίας, στην θεωρία όποιος υπολογιστής κινεί διαδικασίες για να μοιράσει τους πόρους του σε έναν ή παραπάνω πελάτες, είναι server. Έτσι, ενώ η ύπαρξη αρχείων σε ένα μηχάνημα δεν ορίζει το μηχάνημα αυτό ότι είναι εξυπηρετητής, αν χρησιμοποιεί μηχανισμούς για να μοιράσει αυτά τα αρχεία, τότε το μηχάνημα αυτό ορίζεται ως file server (εξυπηρετητής αρχείων). Ομοίως, ένα web server λογισμικό μπορεί να τρέχει σε οποιονδήποτε ικανό υπολογιστή, κατατάσσοντας έτσι ακόμα και έναν προσωπικό υπολογιστή ή ένα laptop στον ρόλο του web server.

Η επικοινωνία μεταξύ πελάτη και εξυπηρετητή γίνεται μέσω ενός τοπικού δικτύου, ή ακόμα και μέσω του Διαδικτύου. Σε μεγάλα δίκτυα όπου ο εξυπηρετητής αναλαμβάνει πολλές εξυπηρετήσεις είναι συνήθως υπολογιστής που διαφέρει ως προς τη σύνθεσή του από άλλους κοινούς υπολογιστές, μιας και οι δυνατότητες του είναι σαφώς αναβαθμισμένες. Κύρια χαρακτηριστικά ενός εξυπηρετητή είναι οι επεξεργαστές που υποστηρίζει και χρησιμοποιεί για την επεξεργασία των δεδομένων που δέχεται, οι γρήγοροι και μεγάλης χωρητικότητας σκληροί δίσκοι αλλά και οι ταχύτατες μνήμες που υποστηρίζει. Συνήθως συνοδεύεται από σύστημα διπλής τροφοδοσίας (dual power supply) και από συσκευή αδιάλειπτης παροχής ενέργειας (UPS), για μεγαλύτερη αξιοπιστία και σιγουριά στις παρεχόμενες υπηρεσίες του.

Η έννοια του server σε υλικό επίπεδο, ορίζει ένα υπολογιστικό μοντέλο που είναι εξειδικευμένο για να έχει αυτόν τον ρόλο. Σε γενικές γραμμές, ένας server θα εκπληρώσει τον σκοπό του καλύτερα από έναν απλό προσωπικό ηλεκτρονικό υπολογιστή. Ένας καλός server καταλαμβάνει τεράστιο χώρο σε σχέση με έναν απλό υπολογιστή, φυλάσσεται υπό ιδανικές συνθήκες θερμοκρασίας – υγρασίας, καθώς επίσης προστατεύονται από κακόβουλο λογισμικό (malware), συντηρείται αμφότερα σε υλικό και σε λογισμικό επίπεδο.

5.1.1 Υλικό

Οι υπολογιστές που χρησιμοποιούνται ως εξυπηρετητές δικτύου πρέπει να μπορούν να δουλεύουν όλη την ημέρα και συνεχώς χωρίς διακοπές. Για το λόγο αυτό οι εταιρείες κατασκευής υπολογιστών και τμημάτων υπολογιστών προσφέρουν ξεχωριστές κατασκευές για εξυπηρετητές. Φυσικά αυτές οι κατασκευές μπορούν να χρησιμοποιηθούν και για προσωπικό υπολογιστή όμως ανεβάζουν το κόστος.

Συνηθισμένα χαρακτηριστικά για το υλικό είναι:

- μητρικές πλακέτες με
  - μεγάλη αντοχή στις θερμοκρασίες
  - ενσωματωμένα συστήματα RAID
  - ενσωματωμένη κάρτα οθόνης μικρής μνήμης και ταχύτητας
  - πιο άνετα κατανεμημένα τα στοιχεία της μνήμης και του επεξεργαστή ώστε να είναι δυνατή η ψύξη και των δύο από ξεχωριστά δυνατά ανεμιστηράκια
  - λιγότερες θύρες επέκτασης
  - περισσότερα ενδεικτικά στοιχεία λειτουργίας
  - δυνατότητα χρήσης μνήμης ECC
- κουτιά
  - βαριά
  - με ειδικό κλείδωμα
  - θέσεις για περισσότερα ανεμιστηράκια
  - ειδικά διαμορφωμένους εσωτερικούς χώρους ώστε να γίνεται πιο καλή κυκλοφορία του αέρα
  - ειδικά κουτιά που μπορούν να ενσωματωθούν σε rack
- δίσκοι
  - με προδιαγραφές για μεγαλύτερο συνεχόμενο χρόνο χρήσης
  - που μπορούν να αποσυνδέονται την ώρα που είναι σε λειτουργία

Δωμάτιο εξυπηρετητών της Google

5.1.2 Λογισμικό

Λειτουργικό σύστημα

Τα περισσότερα λειτουργικά συστήματα έχουν ειδική έκδοση για χρήση ως εξυπηρετητή ή μπορούν να διαμορφωθούν έτσι εκ τον υστέρων εκτελώντας λογισμικό που κάνει την εξυπηρέτηση.

Οι περισσότεροι εξυπηρετητές του διαδικτύου αναλαμβάνουν πολύ δουλειά και είναι κατάλληλα οργανωμένοι με χαρακτηριστικά που τους διαφοροποιούν από απλούς εξυπηρετητές από τα οποία είναι:

- **δυνατότητα λειτουργίας χωρίς την ύπαρξη:**
  - **πληκτρολογίου**
  - **οθόνης**
  - **γραφικού περιβάλλοντος**
  - **κάρτας ήχου**

- **δυνατότητες επιλογής χρόνου επεξεργασίας των διάφορων προγραμμάτων**
- **πρόγραμμα επικοινωνίας με το UPS**
- **δυνατότητα χρήσης περισσότερων του ενός επεξεργαστή**
- **δυνατότητα συν-επεξεργασίας με άλλους υπολογιστές – εξυπηρετητές**
- **περισσότερες δυνατότητες αλλαγής του υλικού και αναβάθμισης του λειτουργικού χωρίς την ανάγκη επανεκκίνησης του συστήματος**
- **περισσότερα συστήματα ασφαλείας**

Προγράμματα

Πολλά σύγχρονα προγράμματα δουλεύουν με τη λογική πελάτη – εξυπηρετητή. Ακόμα και το ίδιο το λειτουργικό σύστημα δουλεύει με αυτήν τη λογική. Τα προγράμματα ζητάνε κάποια ενέργεια και το λειτουργικό σύστημα αναλαμβάνει να τα εξυπηρετήσει εκτελώντας τις λειτουργίες που του ζητήθηκαν. Συνήθως τα περισσότερα προγράμματα εξυπηρετητών απαιτούν και ξεχωριστό υπολογιστή-εξυπηρετητή χωρίς αυτό να είναι πάντα απαραίτητο.

Συνηθισμένοι εξυπηρετητές-προγράμματα σε περιβάλλον γραφείου που μπορεί να εκτελούνται στον ίδιο ή σε ξεχωριστούς υπολογιστές είναι:

- **Εξυπηρετητής αρχείων (file server)**
- **Εξυπηρετητής εκτυπωτών (printer server)**
- **Εξυπηρετητής αντιγράφων ασφαλείας (backup server)**
- **Εξυπηρετητής βάσεων δεδομένων (database server)**
- **Εξυπηρετητής φαξ (fax server)**
- **Εξυπηρετητής διαμεσολαβητή (proxy server)**
- **Εξυπηρετητής ηλεκτρονικού ταχυδρομείου (mail server)**
- **Εξυπηρετητής ήχου (sound server)**
- **Εξυπηρετητής γραφικής απεικόνισης**

Μερικοί εξυπηρετητές όπως εκτυπώσεων, ήχου, γραφικής διεπαφής θεωρούνται αυτονόητο ότι εξυπηρετούν τον ίδιο υπολογιστή, όμως αυτό δεν είναι απαραίτητο, για παράδειγμα ένας υπολογιστής μπορεί να μην εκτυπώνει στον δικό του εκτυπωτή αλλά να τις εκτυπώνει σε άλλο υπολογιστή του δικτύου, όμοιος και με τον εξυπηρετητή που απεικονίζει γραφικές διεπαφές, θα μπορούσε να απεικονίζει τα προγράμματα σε μία οθόνη ενός άλλου υπολογιστή του δικτύου.

Συνηθισμένοι εξυπηρετητές-προγράμματα στο Ίντερνετ είναι:

- Παγκόσμιου Ιστού με το πρωτόκολλο http (http server)
- Domain Name System (DNS server)
- Ηλεκτρονικού ταχυδρμείου (mail server)
- Μεταφοράς αρχείων με το πρωτόκολλο FTP (ftp server)
- irc και instant messaging (irc server)
- Επικοινωνίες φωνής
- streaming audio και video (streaming server)
- Online παιχνίδια

5.1.3 Πραγματικές απαιτήσεις

Κάθε εξυπηρετητής δικτύου έχει διαφορετικές ανάγκες για υλικό. Συνήθως έχει ανάγκη από γρήγορη πρόσβαση στο δίκτυο, όμως ένας εξυπηρετητής φαξ δεν θα υποφέρει τόσο αν δεν έχει γρήγορη κάρτα δικτύου ή αρκετή μνήμη, όσο το να έχει προβληματικό modem. Ένας εξυπηρετητής αρχείων και ένας εξυπηρετητής αντιγράφων ασφαλείας είναι προτιμότερο να έχουν όσο γίνεται πιο αξιόπιστους και πιο ταχείς δίσκους από το να έχουν μεγαλύτερο επεξεργαστή ή περισσότερη και ακριβότερη μνήμη.

Ένας εξυπηρετητής DNS χρειάζεται όσο το δυνατό μεγαλύτερη ασφάλεια και γι' αυτό το λόγο εκείνο που χρειάζεται περισσότερο είναι να τρέχουν όσο το δυνατόν λιγότερα προγράμματα στο λειτουργικό.

5.2 Βάση δεδομένων [13]

Με τον όρο βάση δεδομένων εννοείται μία συλλογή από *συστηματικά μορφοποιημένα* σχετιζόμενα δεδομένα στα οποία είναι δυνατή η ανάκτηση δεδομένων μέσω αναζήτησης κατ' απαίτηση. Ο Αμερικανός επιστήμονας υπολογιστών Τζιμ Γκρέϊ (Jim Gray) έχει γράψει για τις βάσεις δεδομένων: «Όταν οι άνθρωποι χρησιμοποιούν τις λέξεις *βάση δεδομένων*, διατυπώνουν στην ουσία ότι τα δεδομένα πρέπει να αυτοπροσδιορίζονται και να έχουν μια σχηματική δομή. Αυτό ακριβώς περιγράφουν οι λέξεις *βάση δεδομένων*».

Ειδικότερα, στην επιστήμη της πληροφορικής και στην καθημερινή χρήση των ηλεκτρονικών υπολογιστών, με τον όρο *βάσεις δεδομένων* αναφερόμαστε σε οργανωμένες, διακριτές συλλογές σχετιζόμενων δεδομένων ηλεκτρονικά και ψηφιακά αποθηκευμένων, στο λογισμικό που χειρίζεται τέτοιες συλλογές (Σύστημα Διαχείρισης

Βάσεων Δεδομένων, ή DBMS) και στο γνωστικό πεδίο που το μελετά. Πέρα από την εγγενή της ικανότητα να αποθηκεύει δεδομένα, η βάση δεδομένων παρέχει μέσω του σχεδιασμού και του τρόπου ιεράρχησης των δεδομένων, τα αποκαλούμενα συστήματα διαχείρισης περιεχομένου, δηλαδή τη δυνατότητα γρήγορης άντλησης και ανανέωσης των δεδομένων.

Ένας τηλεφωνικός κατάλογος, για παράδειγμα, θεωρείται βάση δεδομένων, καθώς αποθηκεύει και οργανώνει σχετιζόμενα τμήματα πληροφορίας, όπως είναι το όνομα και ο αριθμός τηλεφώνου. Ένα δεύτερο παράδειγμα είναι οι πίνακες δημογραφικών/στατιστικών στοιχείων. Ωστόσο οι σύγχρονες βάσεις δεδομένων υλοποιούνται κυρίως ψηφιακά, με τυποποιημένες μεθόδους σε ηλεκτρονικούς υπολογιστές, ελέγχονται από λογισμικό DBMS και συνιστούν αντικείμενο επιστημονικής και τεχνικής μελέτης από το αντίστοιχο ακαδημαϊκό πεδίο.

Κορυφαίες βάσεις δεδομένων [14]

Όσο περισσότερο μία βάση δεδομένων αναζητείται για τα στοιχεία του, τόσο πιο δημοφιλή θεωρείται η βάση δεδομένων, είναι ένας βασικός δείκτης που συμβολίζει την αξιοπιστία, την ευκολία χρήσης, την δημοτικότητα. Ο πίνακας 1 που ακολουθεί, δημιουργείται από την ανάλυση του πόσο συχνά γίνεται επίσκεψη στις πιο εξελιγμένες γλώσσες που υπάρχουν, μέσω του διαδικτύου. Τα ακατέργαστα δεδομένα πηγάζουν από το Google Trends, ένα Api της Google που παρέχει στατιστικά για τις αναζητήσεις που γίνονται καθημερινά.

	Βάση δεδομένων	Μερίδιο
1	Oracle	36.01%
2	MySQL	22.40%
3	SQL Server	17.85%
4	PostgreSQL	3.15%
5	MongoDB	2.83%

Πίνακας 1 - δημοφιλέστερες βάσεις δεδομένων, 2015

Παγκοσμίως, η Oracle είναι η πιο δημοφιλή βάση δεδομένων, η mongoDB ανέβηκε περισσότερο τα τελευταία 4 χρόνια (2,4%), και η Oracle έχασε το περισσότερο (-3,9%).

Η βάση δεδομένων του παρόν συστήματος

Το RDBMS (Relational Database Management System) που θα χρησιμοποιηθεί για τους σκοπούς της εργασίας, είναι το MySql. Είναι ανοιχτού κώδικα, δημιουργημένο από ένα μεγάλη δυαδικτιακή κοινώτητα και μπορεί να εξυπηρετήσει όλες τις υπηρεσίες που χρειάζονται για το πέρας της ιδέας.

Ο Apache server του συστήματος

Για την ανάγκη του server θα χρησιμοποιηθεί ένας apache server ο οποίος θα υπηρετεί τον php κώδικα που θα εκτελείται.

XAMPP [15]

Το XAMPP είναι ένα πακέτο προγραμμάτων ελεύθερου λογισμικού, λογισμικού ανοικτού κώδικα και ανεξαρτήτου πλατφόρμας το οποίο περιέχει το εξυπηρετητή ιστοσελίδων http Apache, την βάση δεδομένων MySQL και ένα διερμηνέα για κώδικα γραμμένο σε γλώσσες προγραμματισμού PHP και Perl.

Το XAMPP είναι ακρωνύμιο και αναφέρεται στα παρακάτω αρχικά:

- **X - αναφέρεται στο «cross-platform» που σημαίνει λογισμικό ανεξάρτητο πλατφόρμας)**
- **Apache - HTTP εξυπηρετητής**
- **MySQL - Βάση δεδομένων**
- **php**
- **Perl**

Το XAMPP είναι ένα ελεύθερο λογισμικό το οποίο περιέχει ένα εξυπηρετητή ιστοσελίδων το οποίο μπορεί να εξυπηρετεί και δυναμικές ιστοσελίδες τεχνολογίας

PHP/MySQL. Είναι ανεξάρτητο πλατφόρμας και τρέχει σε Microsoft Windows, Linux, Solaris, and Mac OS X και χρησιμοποιείται ως πλατφόρμα για την σχεδίαση και ανάπτυξη ιστοσελίδων με την τεχνολογίες όπως PHP, JSP και Servlets.

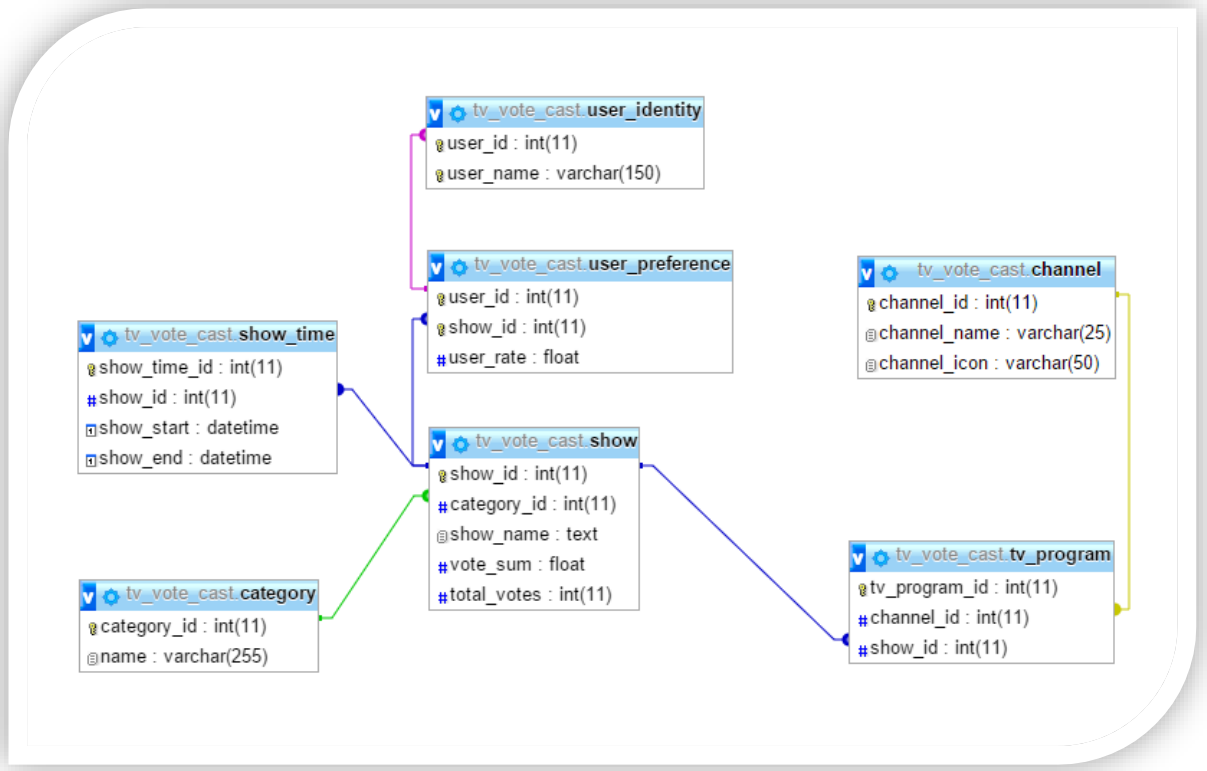
Το XAMPP προϋποθέτει μόνο τα λογισμικά συμπίεσης αρχείων zip, tar, 7z ή exe κατά την διάρκεια της εγκατάστασης. Το XAMPP έχει δυνατότητα αναβάθμισης σε νέες εκδόσεις του εξυπηρετητή ιστοσελίδων http Apache, της βάσης δεδομένων MySQL, της γλώσσας PHP και Perl. Το XAMPP συμπεριλαμβάνει επίσης τα πακέτα OpenSSL και το phpMyAdmin.

Επίσημα οι σχεδιαστές του XAMPP προόριζαν το λογισμικό ως εργαλείο ανάπτυξης και δοκιμής ιστοσελίδων τοπικά στον υπολογιστή χωρίς να είναι απαραίτητη η σύνδεση στο διαδίκτυο. Για να είναι δυνατή η χρήση του, πολλές σημαντικές λειτουργίες ασφάλειας έχουν απενεργοποιηθεί. Στην πράξη το XAMPP ορισμένες φορές χρησιμοποιείται και για την φιλοξενία ιστοσελίδων. Υπάρχει ειδικό εργαλείο το οποίο περιέχεται στο XAMPP για την προστασία με κωδικό των σημαντικών μερών. Το XAMPP υποστηρίζει την δημιουργία και διαχείριση βάσεων δεδομένων τύπου MySQL και SQLite. Όταν το XAMPP εγκατασταθεί στον τοπικό υπολογιστή διαχειρίζεται τον localhost ως ένα απομακρυσμένο κόμβο, ο οποίος συνδέεται με το πρωτόκολλο μεταφοράς αρχείων FTP. Η σύνδεση στον localhost μέσω του FTP μπορεί να γίνει με το όνομα χρήστη «newuser» και το κωδικό «wampp». Για την βάση δεδομένων MySQL υπάρχει ο χρήστης «root» χωρίς κωδικό πρόσβασης.

Λόγο των παραπάνω, τα XAMPP θα χρησιμοποιηθεί για να εξυπηρετεί το Server side μέρος ή αλλιώς το back end σύστημα.

5.3 Σχήμα βάσης προγράμματος Tv Vote Cast

Το παρακάτω σχήμα, παρουσιάζει τους πίνακες της βάσης και τις μεταξύ τους σχέσεις.



Εικόνα 8 - Σχήμα βάσης δεδομένων Tv vote cast

Πίνακες βάσης ονομαστικά

1	category
2	channel
3	show
4	show_time
5	tv_program
6	user_identity
7	user_preferences
6	**Program_view**

Πίνακας category

1. category

Column	Type	Null	Extra
category_id	int(11)	no	auto_increment
name	varchar(255)	no	

Ο πίνακας "category" περιέχει όλες της κατηγορίες, το category_id χρησιμοποιείται ως πρωτεύον κλειδί του πίνακα και ως ξένο κλειδί στον πίνακα "show". Η σχέση των πινάκων "category" - "show" είναι ένα προς πολλά.

Πίνακας channel

2. channel

Column	Type	Null	Extra
channel_id	Int(11)	No	auto_increment
channel_name	Varchar(25)	No	
channel_icon	varchar(50)	Yes	

Ο πίνακας channel περιέχει την λίστα των καναλιών, καθώς επίσης και το αντίστοιχο όνομα του εικονιδίου το οποίο εμφανίζεται στην εφαρμογή. Το channel_id είναι το πρωτεύον κλειδί του πίνακα το οποίο χρησιμοποιείται σαν ξένο κλειδί στον πίνακα "tv_program". Η σχέση των πινάκων "channel" - "tv_program" είναι ένα προς πολλά.

Πίνακας show

3. show

Column	Type	Null	Extra	Links to
show_id	int(11)	No	auto_increment	
category_id	int(11)	No		category→category_id
show_name	text	No		
vote_sum	float	Yes		
total_votes	int(11)	Yes		

Ο πίνακας "show" περιέχει τους τίτλους των προβολών καθώς επίσης και τις κριτικές των χρηστών. Έχει πρωτεύον κλειδί το show_id το οποίο είναι ξένο κλειδί στον πίνακα "show_time" που περιέχει τις πληροφορίες για την ώρα της προβολής. Η σχέση του πρώτου με τον δεύτερο είναι ένα προς πολλά, αυτό σημαίνει ότι μία προβολή μπορεί να έχει ώρα παρουσίασης παραπάνω από μία, με την λογική ότι για παράδειγμα οι ειδήσεις ενός καναλιού είναι ίδιο show αλλά προβάλλεται πολλές φορές ανά την ημέρα, επίσης μία ταινία έχει ημέρα προβολής πολλές φορές μέσα στον χρόνο, παρόλα αυτά είναι η ίδια ταινία άρα της αντιστοιχεί ίδιο show_id. Το show_id είναι ξένο κλειδί στον πίνακα "tv_program", η σχέση του πρώτου με τον δεύτερο είναι ένα προς πολλά. Τέλος, το vote_sum και total_votes είναι πεδία που περιγράφουν την αξιολόγηση της εκπομπής, το άθροισμα των αξιολογήσεων και το άθροισμα των ψηφοφοριών αντίστοιχα. Η διαίρεση του πρώτου από το δεύτερο αποτελούν την μέση αξιολόγηση με μέγιστο το 5.

Πίνακας show_time

4. show_time

Column	Type	Null	Extra	Links to
show_time_id	int(11)	No	auto_increment	
show_id	int(11)	No		show→show_id
show_start	datetime	No		
show_end	datetime	No		

Ο πίνακας "show_time" περιέχει τις ώρες της κάθε προβολής όπως περιγράφονται στις κολόνες show_start και show_end. Πρωτεύον κλειδί είναι το show_time_id, ξένο κλειδί το show_id που αντιστοιχεί τον εν λόγο πίνακα με τον πίνακα "show".

Πίνακας tv_program

5. tv_program

Column	Type	Null	Extra	Links to
tv_program_id	int(11)	No	auto_increment	
channel_id	int(11)	No		channel→channel_id
show_id	int(11)	No		show→show_id

Ο πίνακας *"tv_program"* έχει σκοπό να ενώσει τους προαναφερθέντες πίνακες και να περιέχει τις οντότητες του προγράμματος. Πρωτεύων κλειδί είναι το tv_program_id, channel_id είναι το ξένο κλειδί του πίνακα *"channel"*, show_id είναι το ξένο κλειδί του πίνακα *"show"*.

Πίνακας user_identity

6. user_identity

Column	Type	Null	Extra	Links to
user_id	int(11)	No	auto_increment	
user_name	int(11)	No		

Ο πίνακας *"user_identity"* είναι ο πίνακας που καταγράφει τα στοιχεία των χρηστών, περιέχει τον ειδικό δείκτη user_id και το όνομα του χρήστη. Αν η εφαρμογή προοριζόταν για παραγωγή, ο πίνακας αυτός θα μπορούσε να περιελάβανε και άλλα στοιχεία για τον χρήστη όπως διεύθυνση, τηλέφωνο, email· όλα όσα ζητάνε οι εφαρμογές που κυκλοφορούν σήμερα. Το Tv Vote Cast είναι μία εφαρμογή που δεν θέλει να προκαλέσει απόγνωση στον χρήστη ζητώντας πολλά προσωπικά στοιχεία. Παρόλα αυτά εάν η εφαρμογή εξελισσόταν και έδινε στον χρήστη αιτιολογία για ποιόν λόγο τα ζητάει, τότε όλα αυτά τα στοιχεία θα εισέρχονταν στον πίνακα "*user_identity*". Το user_id αποτελεί ξένο κλειδι του πίνακα "*user_preferences*".

Πίνακας user_preferences

7. user_preferences

Column	Type	Null	Extra	Links to
user_id	int(11)	No		user_identity→user_id
show_id	int(11)	No		show→show_id
user_rate	float	No		show→show_id

Ο πίνακας "*user_preferences*" καταγράφει τις προτιμήσεις των χρηστών για κάθε εκπομπή που έχουν καταθέσει κριτική. Σε επίπεδο βάσης, ο πίνακας είναι ενδιάμεσος του πίνακα "*show*" και "*user_identity*". Κλειδί του πίνακα είναι ο συνδυασμός του user_id και του show_id, δηλαδή ένας χρήστης μπορεί να έχει αξιολογήσει πολλές εκπομπές, μπορεί επίσης να έχει ψηφίσει την ίδια εκπομπή πολλές φορές, αλλά η βάση κρατάει την τελευταία καταχώρηση μόνο. Ως εκ τούτου, σε ένα user_id δεν μπορεί να υπάρχει ίδιο show_id πάνω από μία φορά. Τα δεδομένα που καταγράφονται σε αυτόν τον πίνακα προσπελάυνονται κατά την εκτέλεση του συνεργατικού φιλτραρίσματος για την εύρεση προτεινόμενων εκπομπών.

Πίνακας program_view

6. program_view

Column	Type	Null
tv_program_id	int(11)	No
channel_id	int(11)	No
channel_name	varchar(25)	No
channel_icon	varchar(50)	Yes
show_id	int(11)	No
show_name	text	No
show_start	varchar(10)	Yes
show_end	varchar(10)	Yes
vote_rating	double	No

Στην θεωρία των βάσεων δεδομένων, ένα view είναι το σύνολο των αποτελεσμάτων ενός αποθηκευμένου ερωτήματος στα δεδομένα, στα οποία ένας χρήστης μπορεί να ρωτήσει σαν να ήταν πίνακας. Αυτό το προκαθορισμένο ερώτημα φυλάσσεται στο λεξικό της βάσης. Αντίθετα με τους συνηθισμένους πίνακες της σχεσιακής βάσης, ένα view δεν αποτελεί κομμάτι του φυσικού σχήματος, είναι ένας εικονικός πίνακας ο οποίος υπολογίζεται δυναμικά από τα δεδομένα της βάσης όταν ζητηθεί. Αλλαγές στα δεδομένα των πινάκων που σχηματίζουν το view αντικαθρεφτίζονται αμέσως σε αυτό.

Ο program_view είναι ένα view των πινάκων "tv_program", "channel", "show" και "show_time". Αυτό σημαίνει ότι στην πραγματικότητα δεν υπάρχει, είναι το αποτέλεσμα ενός select το οποίο επιλέγει όλα τα δεδομένα που χρειάζεται για να παρουσιαστεί στην εφαρμογή μία εκπομπή. Το view χρησιμεύει στο να είναι πιο εύκολη και πιο

γρήγορη η διαδικασία της επιλογής δεδομένων που χρειάζονται για την παρουσίαση του εικαστικού.

Το Select query που παράγει τον πίνακα program_view

```
SELECT
    `tv_program`.`tv_program_id` AS `tv_program_id`,
    `tv_program`.`channel_id` AS `channel_id`,
    `tv_program`.`show_id` AS `show_id`,
    `channel`.`channel_icon` AS `channel_icon`,
    `show`.`show_name` AS `show_name`,
    `channel`.`channel_name` AS `channel_name`,
    date_format(`show_time`.`show_start`,'%k:%i') AS `show_start`,
    date_format(`show_time`.`show_end`,'%k:%i') AS `show_end`,
    ifnull((`show`.`vote_sum` / `show`.`total_votes`),0) AS `vote_rating`,
    `category`.`category_id` AS `category_id`,
    `category`.`name` AS `category_name`
FROM ((((`tv_program` left join `show_time` on((`tv_program`.`show_id` = `show_time`.`show_id`)))
    left join `show` on((`tv_program`.`show_id` = `show`.`show_id`)))
    left join `channel` on((`tv_program`.`channel_id` = `channel`.`channel_id`)))
    left join `category` on((`show`.`category_id` = `category`.`category_id`)))
order by cast(`show_time`.`show_start` as time)
```

6. ΑΛΓΟΡΙΘΜΟΣ ΣΥΝΕΡΓΑΤΙΚΟΥ ΦΙΛΤΡΑΡΙΣΜΑΤΟΣ

Τα συστήματα συστάσεων είναι ένα σημαντικό μέρος του οικοσυστήματος της πληροφορικής και του ηλεκτρονικού εμπορίου. Αντιπροσωπεύουν μία ισχυρή μέθοδο για να φιλτράρεται μεγάλος όγκος πληροφοριών και προϊόντων. Σχεδόν δύο δεκαετίες έρευνας για το συνεργατικό φιλτράρισμα έχουν οδηγήσει σε ένα ποικίλο αριθμό αλγορίθμων και μία πλούσια συλλογή εργαλείων για την αξιολόγηση της απόδοσής τους. Η έρευνα κινείται προς την κατεύθυνση βαθύτερης κατανόησης για το πώς τα συστήματα συστάσεων μπορούν να ενσωματωθούν σε συγκεκριμένους τομείς. Λόγο της ύπαρξης πολλών ανθρώπινων προσωπικοτήτων, ο αλγόριθμος συστάσεων δεν είναι ένα πρόβλημα που έχει μόνο μία λύση. Ανάλογα με το έργο, τις ανάγκες, τον τομέα, και άλλα έχει να αντιμετωπίσει μοναδικά προβλήματα για να κάνει μία σύσταση, για αυτό η αξιολόγηση είναι σχεδόν πάντα διαφορετική ανάλογα με την περίπτωση. Για να είναι αποτελεσματική η ανάπτυξη ενός τέτοιου αλγορίθμου, πρέπει να γίνει προσεκτική ανάλυση των δυνητικών χρηστών και των στόχων τους. Με βάση αυτή την ανάλυση, οι σχεδιαστές του συστήματος έχουν μια σειρά από επιλογές για την επιλογή του αλγορίθμου και την ενσωμάτωσή του στο περιβάλλον του χρήστη.

Καθημερινά, πλημμυριζόμαστε από διλήμματα και επιλογές. Τί να φορέσουμε, τί να αγοράσουμε, τί να διαβάσουμε, τί εκπομπή στην τηλεόραση να δούμε. Το πλήθος των επιλογών που έχουμε και πρέπει να αποφασίσουμε σε αρκετές περιπτώσεις είναι τεράστιο. Για παράδειγμα, το Netflix έχει πάνω από 17.000 ταινίες στην συλλογή του, το Amazon.com έχει περισσότερα από 410.000 βιβλία. Οι άνθρωποι στηρίζονται αρκετά σε συστάσεις και αναφορές είτε απλών συνανθρώπων, είτε ειδικών εμπειρογνωμόνων για την λήψη αποφάσεων ή για την εύρεση νέου υλικού. Συζητούν για τις προτιμήσεις τους, διαβάζουν κριτικές στις εφημερίδες, στα περιοδικά, στο διαδίκτυο, ρωτούν τον βιβλιοθηκάριο για ένα βιβλίο, ή ανοίγουν την τηλεόραση και παρακολουθούν ό,τι τυγχάνει να παίζει.

Οι μέθοδοι συνεργατικών συστάσεων[16] νέων πραγμάτων έχουν τα όριά τους, ειδικά για εύρεση καινούριων πληροφοριών. Μπορεί να υπάρχει μία ταινία ανεξάρτητης παραγωγής, ή ένα βιβλίο το οποίο είναι ενδιαφέρον, αλλά κανείς από τον κύκλο γνωριμιών να μην γνωρίζει. Μπορεί να υπάρχει μία μπάντα από άλλη πόλη, ακόμα και χώρα της οποία η μουσική δεν έχει ακουστεί στους τοπικούς κριτές. Συστήματα που βασίζονται σε υπολογιστές παρέχουν την ευκαιρία να επεκταθεί το σύνολο των ανθρώπων από τους οποίους οι χρήστες μπορούν να αποκτήσουν συστάσεις. Επίσης δίνουν την δυνατότητα στον παρατηρητή να γνωρίσει την ιστορικότητα αυτού που

δήλωσε την προτίμηση, κάτι το οποίο παρέχει μεγαλύτερη λεπτομέρεια στα κριτήρια της τελικής επιλογής.

Υπήρξε μεγάλη έρευνα πάνω στο αντικείμενο τα τελευταία 20 χρόνια για το πώς μπορούν να αυτοματοποιηθούν οι συνεργατικές συστάσεις προς τους ανθρώπους, και έχει προταθεί μία μεγάλη ποικιλία μεθόδων. Πρόσφατα ένα εγχειρίδιο[17] για τα συστήματα συνεργατικών συστάσεων δόθηκε στην δημοσιότητα, παρέχοντας συζητήσεις σε βάθος για μεθόδους που κάνουν συστάσεις στους ανθρώπους. Η έρευνα ωστόσο επικεντρώνεται στον αλγόριθμο του συνεργατικού φιλτραρίσματος, μία κατηγορία μεθόδων που συνιστά στοιχεία, με βάση τις προτιμήσεις άλλων χρηστών στα εν λόγω στοιχεία.

Εκτός από το ακαδημαϊκό ενδιαφέρον, τα συστήματα συνεργατικών συστάσεων βλέπουν μεγάλο ενδιαφέρον από την βιομηχανία. Η Amazon χρησιμοποιεί συνεργατικό φιλτράρισμα εδώ και μία δεκαετία, για να προτείνει τα προϊόντα της στους πελάτες. Το Netflix σημείωσε μεγάλες εισοδηματικές αυξήσεις που ξεπέρασαν το ένα εκατομμύριο δολάρια στην υπηρεσία ενοικίασης ταινιών που παρέχει παγκοσμίως μέσω του Netflix Prize.

Υπάρχει αυξανόμενο ενδιαφέρον για τα προβλήματα γύρω από τον υπολογισμό των συνεργατικών συστάσεων, αλγόριθμοι για την κατανόηση και την πρόβλεψη των προτιμήσεων των χρηστών δεν υπάρχουν ετοιμοπαράδοτοι για όλες τις περιπτώσεις. Ένα σύστημα συνεργατικών συστάσεων πρέπει να αλληλοεπιδράσει με τον χρήστη, ώστε και να μάθει τις προτιμήσεις του, αλλά και να προσφέρει συστάσεις σε τρίτους. Πρέπει να έχει ακριβή δεδομένα από τα οποία θα υπολογιστούν οι συνεργατικές συστάσεις και οι προτιμήσεις, το οποίο οδηγεί στην ανάγκη ύπαρξης αξιόπιστων δεδομένων και την ανάγκη μείωσης του θορύβου. Οι χρήστες επίσης έχουν ποικίλους στόχους και ποικίλες ανάγκες, από βασικές ανάγκες για εύρεση πληροφοριών σε πιο περίπλοκες επιθυμίες, όπως επιθυμία προστασίας της ιδιωτικής ζωής.

6.1 Ιστορική αναδρομή στα Συστήματα συνεργατικών συστάσεων [18]

Η δυνατότητα των υπολογιστών να παρέχουν συμβουλευτικές συστάσεις, αναγνωρίστηκε σχετικά νωρίς στην ιστορία των υπολογιστών. Ένα μεγάλο βήμα έγινε το 1979 με την δημιουργία ενός ηλεκτρονικού βιβλιοθηκάριου, ήταν μία πολύ πρωτόγονη προσπάθεια. Έκανε ομαδοποίηση των χρηστών με βάση κάποια στερεότυπα κριτήρια και χρησιμοποιούσε στατική πληροφορία βάση κάποιων στερεότυπων βιβλίων, για να βρει συστάσεις και να τις προτείνει. Παρόλη την απλότητα του τότε συστήματος, δεν παύει

να είναι μία πρωταρχική προσπάθεια για την δημιουργία των πρώτων υπολογιστών που προτείνουν "αντικείμενα".

Την δεκαετία του 90, ο αλγόριθμος συνεργατικού φιλτραρίσματος ξεκίνησε να ίσταται σαν λύση για την αντιμετώπιση του προβλήματος του μεγάλου όγκου δεδομένων. Το Tapestry ήταν ένα εγχειρίδιο για τα συστήματα συνεργατικού φιλτραρίσματος: έδινε την δυνατότητα στους χρήστες να ρωτήσουν για αντικείμενα άλλων χρηστών, τα οποία βρισκόταν μέσα σε ένα πληροφοριακό σύστημα, όπως το e-mail μίας εταιρείας. Βασιζόταν στην άποψη και την δράση των άλλων χρηστών, πχ. "Δώσε όλα τα μηνύματα που προωθήθηκαν από τον Γιάννη".

Αυτοματοποιημένα συστήματα συνεργατικών συστάσεων σύντομα ακολούθησαν, με αυτοματοποιημένες διαδικασίες εντόπιζαν σχετικές απόψεις και σύμφωνα με το σύνολο τους παρείχε συστάσεις. Μία αμερικάνικη εταιρεία χρησιμοποίησε αυτήν την τεχνική για να αναγνωρίζει ηλεκτρονικά άρθρα που πιθανόν να απασχολούν άλλους χρήστες. Οι χρήστες το μόνο που έπρεπε να κάνουν ήταν να καταθέτουν την κριτική τους ή να κάνουν άλλες κινήσεις παρακολούθησης που δηλώνουν ενδιαφέρον. Το σύστημα ένωνε τα δεδομένα εισόδου από τους χρήστες για να υπολογίσει τα αποτελέσματα σε προσωπικό επίπεδο. Σε αυτά τα συστήματα ο τρέχων χρήστης δεν μαθαίνει πληροφορίες για την άποψη των υπόλοιπων χρηστών, ούτε καν χρειάζεται τέτοια πληροφορία για να αποσπάσει συστάσεις.

Κατά την διάρκεια αυτής της περιόδου, τα συστήματα συστάσεων και το συνεργατικό φιλτράρισμα έγινε ένα θέμα αυξανόμενου ενδιαφέροντος, ειδικά σε θέματα αλληλεπίδρασης του ανθρώπου με την μηχανή, σε θέματα machine learning, και σε θέματα αναζήτησης πληροφορίας. Αυτό το ενδιαφέρον παρήγαγε έναν αισθητό αριθμό συστημάτων συνεργατικών συστάσεων σε ποικίλους τομείς, όπως το Ringo που προτείνει μουσική, το BellCore το οποίο προτείνει βίντεο, και το Jester αστεία. Έξω από τον τομέα της πληροφορικής, ο τομέας των προωθήσεων (marketing) έχει συνειδητοποιήσει ότι οι συστάσεις έχουν την ικανότητα να ανεβάζουν τις πωλήσεις και να βελτιώνουν την εμπειρία του πελάτη.

Στα τέλη της δεκαετίας του 90, ξεκίνησαν να αναδύονται συστήματα συνεργατικών συστάσεων για διαφημιστικούς σκοπούς. Ίσως η πιο γνωστή περίπτωση τέτοιας τεχνολογίας, είναι το Amazon.com. Βασιζόμενη στο ιστορικό του περιηγητή (browser), στο ιστορικό των αγορών, και τα αντικείμενα που κοιτάει ο χρήστης κατά την περιήγησή του σε διαδικτυακά καταστήματα, προτείνει παρόμοια προϊόντα. Από τότε που η Amazon χρησιμοποίησε αυτήν την μέθοδο, ο αλγόριθμος υιοθετήθηκε και από

άλλους οργανισμούς του διαδικτύου. Ένα σημαντικό κίνητρο είναι η αύξηση των πωλήσεων, ένα αντικείμενο θα αγοραστεί από έναν πελάτη εάν του έχει προταθεί, αν δεν το γνωρίζει δεν υπάρχει καμία πιθανότητα να γίνει αγορά. Αρκετές εταιρείες, όπως η NetPerceptions και η Strands, έχουν δημιουργηθεί με αντικείμενο τις τεχνολογίες συνεργατικών συστάσεων για online πωλήσεις.

Πλέον, έχουν δημιουργηθεί πολλές τεχνικές που προσθέτουν στον αλγόριθμο συνεργατικού φιλτραρίσματος, ανάκτηση δεδομένων με βάση το περιεχόμενο, μέθοδοι Μπεϋζιανων διεπαφών (*bayesian inference*[19]), είναι μερικές από τις τεχνικές που έχουν ανακαλυφθεί από έρευνες. Στο Tv Vote Cast, όταν γίνει αξιολόγηση μίας εκπομπής, οι εκπομπές που είναι υποψήφιες για να προταθούν, ανήκουν στην ίδια κατηγορία με αυτή που ανήκει η τρέχων εκπομπή, δηλαδή γίνεται χρήση της πρώτης τεχνικής από τις προαναφερθέν.

Κατά την διάρκεια αντιμετώπισης πολλών περιπτώσεων, δημιουργήθηκαν υβριδικά συστήματα, τα οποία ενώνουν πολλούς αλγόριθμους διαφορετικού σκοπού, για να συγκροτήσουν ένα συγκεντρωτικό αλγόριθμο συστάσεων. Παρ' όλα αυτά, ο αλγόριθμος συνεργατικού φιλτραρίσματος έχει έναν βασικό κορμό, ο οποίος είτε μόνος του, είτε υποβοηθούμενος από περιφερειακούς αλγορίθμους είναι αποτελεσματικός και για αυτό δεν αλλάζει.

Οι έρευνες που γίνονταν πάνω στους αλγόριθμους συστάσεων μάζεψαν αρκετό ενδιαφέρον το 2006, όταν το Netflix ξεκίνησε το Netflix Prize με σκοπό να βελτιώσει την αποδοτικότητα των προτεινόμενων ταινιών. Το Netflix Prize ήταν ένας διαγωνισμός που είχε στόχο την δημιουργία ενός καινούριου αλγόριθμου που θα κέρδιζε τον παλιό (αλγόριθμο CineMatch) κατά 10%. Ο διαγωνισμός πυροδότησε έντονο ενδιαφέρον σε ακαδημαϊκά ιδρύματα αλλά και σε χομπίστες. Το έπαθλο του ενός εκατομμυρίου δείχνει πόσο αξία δίνουν οι πωλητές στο να παρέχουν εύστοχες συστάσεις.

6.2 Υλοποίηση συνεργατικού φιλτραρίσματος

Σε αυτό το άρθρο θα γίνει εξήγηση υλοποίησης συνεργατικού φιλτραρίσματος. Ο αλγόριθμος αυτός φιλτράρει πληροφορίες για λογαριασμό ενός χρήστη που βρίσκεται σε ένα σύνολο πολλών. Χρήστες με παρόμοιο ιστορικό προφίλ, μπορεί να έχουν κοινά ενδιαφέροντα. Η συλλογή των δεδομένων σε γενικές γραμμές μπορεί να γίνει είτε ρητά

είτε σιωπηρά, στην περίπτωση του Tv Vote Cast δεν υπάρχει ανάγκη να γίνεται ρητά. Τα δεδομένα που θα καταγραφούν δεν θα χρησιμοποιηθούν για να παρουσιαστούν οι προτιμήσεις των χρηστών σε αντιστοιχία με αυτούς, η απόφαση αυτή είναι τελεσίδικη και υπόσχεται τον πλήρη σεβασμό προς τα προσωπικά δεδομένα των ανθρώπων που συμμετάσχουν στην εφαρμογή. Τα ονόματα των χρηστών θα χρησιμοποιηθούν μόνο για να παρουσιάσουν τα αντικείμενα που έχουν στην κατοχή τους και τί πιστεύουν για αυτά. Με τα αντικείμενα αυτά θα γίνει ένας διαγωνισμός ποιο είναι το πλησιέστερο στο αντικείμενο που κρατάει ο χρήστης που ρωτάει. Στην εφαρμογή Tv Vote Cast είναι ο άνθρωπος που κρατάει το κινητό ή αλλιώς *τρέχων χρήστης*, περιηγείται στις λίστες των εκπομπών και δίνει την ψήφο του σε ορισμένες από αυτές. Εάν οι ίδιες ταινίες που έχουν δει δύο χρήστες είναι παρόμοια βαθμολογημένες, τότε υπάρχει μεγάλη πιθανότητα ένας τυχαίος χρήστης να έχει τις ίδιες προτιμήσεις με τον τρέχων χρήστη. Ο τρέχων χρήστης πριν αποφασίσει τελικά ποιος του ταιριάζει, κάνει την ίδια σύγκριση με όλους τους χρήστες και αξιολογεί ποιος του ταιριάζει περισσότερο. Αυτός που έχει την μεγαλύτερη αξιολόγηση είναι αυτός που θα κάνει την πρόταση του για το ποια ταινία του άρεσε περισσότερο. Από το παράδειγμα των ταινιών φαίνεται ξεκάθαρα, δεδομένου ότι μία εκπομπή βρίσκεται σε κατηγορία εκπομπών (πχ. Ενημέρωση, τηλεοπτικά παιχνίδια, αθλητικές εκπομπές) η συμμετοχή στους διαγωνισμούς θα απαρτίζεται μόνο από εκπομπές ίδιας κατηγορίας. Έτσι παράγεται το αποτέλεσμα, να κάνει συστάσεις ο ένας χρήστης στον άλλο χωρίς να γνωριστούν οι προσωπικότητες.

Το πρώτο πράγμα που πρέπει να πραγματοποιηθεί είναι η συλλογή των προτιμήσεων των χρηστών. Τα στοιχεία αυτά καταγράφονται στην βάση δεδομένων στους πίνακες *user_identity*, *user_preference*, και *show*. Η συλλογή των στοιχείων έχει την παρακάτω μορφή.

User / Show	Nerit news	World party	Just cooking	Champions league show
Ανδριάνα	3.5	4	1.5	3.5
Δήμος	1.5	5	3.5	3.5
Μαρίνος	4.5	3	2.5	3

Πίνακας 2 - Συμβολικός πίνακας δεδομένων

Η συλλογή των δεδομένων που απαρτίζουν τον πίνακα 2 δημιουργείται όταν ένας χρήστης (αριστερή κάθετη κολόνα - User) βαθμολογεί μία εκπομπή (πάνω οριζόντια γραμμή - show), μέσω της εφαρμογής Tv Vote Cast.

Αφού έχει γίνει η συλλογή των δεδομένων, πρέπει μία μέτρηση να καθορίσει πόσο όμοιες είναι προτιμήσεις τους. Για να γίνει αυτή η μέτρηση πρέπει να γίνει σύγκριση για κάθε χρήστη με βάση κοινό μέτρο. Υπάρχουν αρκετές μαθηματικές λειτουργίες που υπολογίζουν το κοινό μέτρο, για το έργο έχει αποφασιστεί να ακολουθεί την Ευκλείδεια απόσταση (σ.σ. εκτός ορίων εργασίας η μαθηματική ανάλυση). Η βασική ιδέα πίσω από τους υπολογισμούς είναι ότι όσο πιο κοινές βαθμολογίες έχουν δύο χρήστες, τόσο πιο κοντά βρίσκονται στις προτιμήσεις και στα ενδιαφέροντα.

Για το ποιόν από όλους τους υποψήφιους πρέπει να εμπιστευτεί ο τρέχων χρήστης είναι το επόμενο πρόβλημα. Θα πρέπει το κοινό μέτρο να χρησιμοποιηθεί σε όλους τους χρήστες και να δώσει την αύξουσα λίστα των αξιολογήσεων. Αυτά γίνονται με κώδικα στο αρχείο *recommend.php* **στην** *function similarityDistance*. **Το πλήθος των συγκριτικών επαναλήψεων μειώνει την αξία της μονάδας μέτρησης των αποστάσεων. Έτσι ο αλγόριθμος συνεργατικού φιλτραρίσματος (από χρήστη προς χρήστη), γνωρίζει ποιος είναι ο νικητής του διαγωνισμού που θα προτείνει την καλύτερή εκπομπή προς τον τρέχων χρήστη.**

Όταν τα δεδομένα έχουν τεράστιο όγκο, όπως στην περίπτωση της Amazon, στην οποία εκατομμύρια χρήστες και αντικείμενα συγκρίνονται μεταξύ τους, το κόστος αξιολόγησης είναι τεράστιο. Μια εναλλακτικές τεχνική του αλγορίθμου έχει δημιουργηθεί ώστε να αποφευχθούν οι περιορισμοί, η τεχνική ονομάζεται φιλτράρισμα με βάση τα αντικείμενα. Η τεχνική αυτή δίνει καλύτερα αποτελέσματα υπό προϋποθέσεις και μπορεί να γίνει πριν ο χρήστης ρωτήσει το συνεργατικό φιλτράρισμα με βάση τον χρήστη για τα τελικά αποτελέσματα.

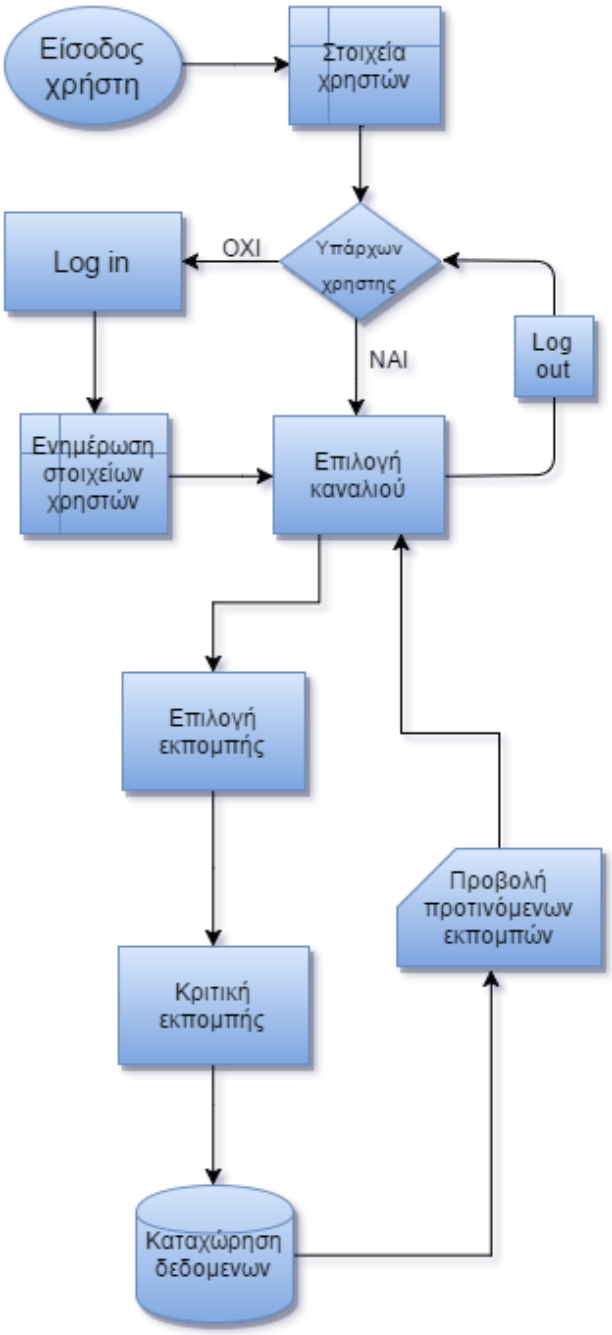
7. ΠΡΟΓΡΑΜΜΑΤΙΣΤΙΚΗ ΑΝΑΠΤΥΞΗ Tv Vote Cast

Στο κεφάλαιο αυτό θα γίνει ανάλυση πώς έγινε η ανάπτυξη της εφαρμογής Tv Vote Cast για κινητή συσκευή με λειτουργικό android.

Εικόνα 9 - Εμπορικό σήμα Tv vote cast (η διπλωματική δεν διακατέχει τα δικαιώματα του εικονιδίου)

7.1 Διάγραμμα ροής

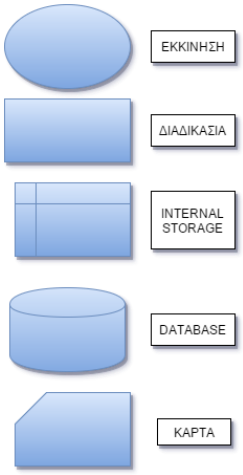
Το Tv Vote Cast αποτελείται από τρείς βασικές κινήσεις. Στην πρώτης καταχωρείται ο χρήστης, και εμφανίζεται η λίστα με τα κανάλια και τα αντίστοιχα εικονίδιά τους. Η δεύτερη περιέχει την εμφάνιση της λίστα με τις εκπομπές που έχει το κανάλι που επιλέχθηκε, παρουσιάζεται επίσης η αξιολόγηση της εκάστοτε εκπομπής από τους άλλους χρήστες. Η τρίτη οθόνη, περιέχει μία φόρμα η οποία δίνει την δυνατότητα στον χρήστη να συμπληρώσει με πόσα αστέρια θέλει να αξιολογήσει την εκπομπή που επέλεξε από την προηγούμενη οθόνη. Αφού πατήσει το κουμπί "vote" εμφανίζεται η οθόνη με τις προτεινόμενες εκπομπές. Οι προτεινόμενες εκπομπές είναι τρείς , αποτέλεσμα του συνεργατικού φιλτραρίσματος. Η εικόνα 10 παρουσιάζει το διάγραμμα ροής της εφαρμογής Tv Vote Cast στην κινητή συσκευή που κρατάει ο χρήστης στα χέρια του.



Εικόνα 10 - διάγραμμα ροής Tv vote cast

Πρότυπα σύμβολα διαγράμματος ροής

Το πρότυπο που χρησιμοποιείται είναι με βάση το EdrawSoft®[20]

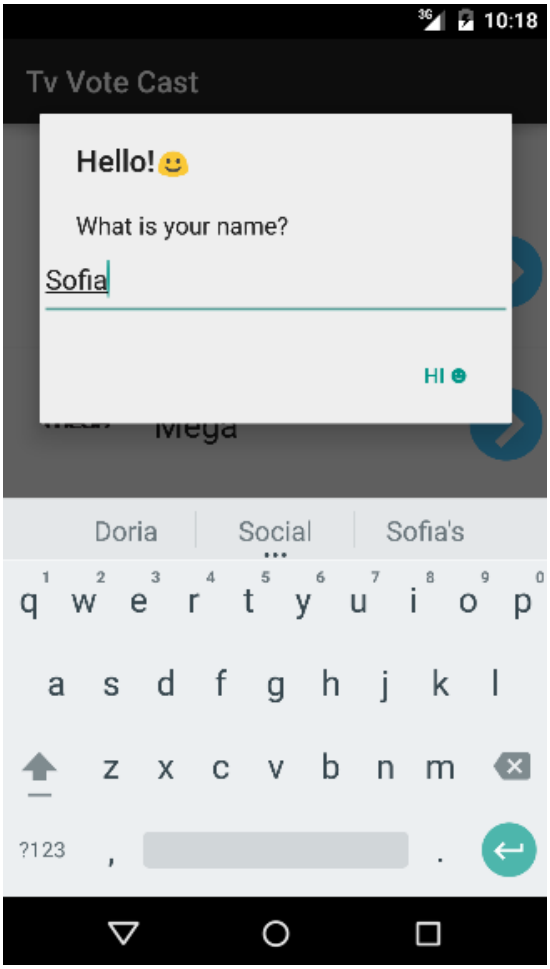


Εικόνα 11 - πρότυπα σύμβολα

7.2 Αναλυτική επεξήγηση κώδικα Android

Έλεγχος για την καταχώρηση χρήστη

Κατά την παρακάτω διαδικασία πραγματοποιούνται τα εξής: πρώτον ο έλεγχος για το αν υπήρξε στο παρελθόν εγγεγραμμένος χρήστης της εφαρμογής. Δηλαδή, έλεγχος αν την τελευταία φορά που κάποιος χρησιμοποίησε την εφαρμογή, είχε καταχωρήσει το όνομά του. Εάν υπήρξε τέτοιος χρήστης, η εφαρμογή τον θεωρεί ως τρέχων χρήστη. Στην περίπτωση που δεν υπήρξε χρήστης στον παρελθόν, τότε η εφαρμογή προχωράει στην δεύτερη λειτουργία της εν λόγο διαδικασίας, που είναι η "γνωριμία" με τον χρήστη μέχρι να γίνει πελάτης.



Εικόνα 12 - Καρτέλα Log in

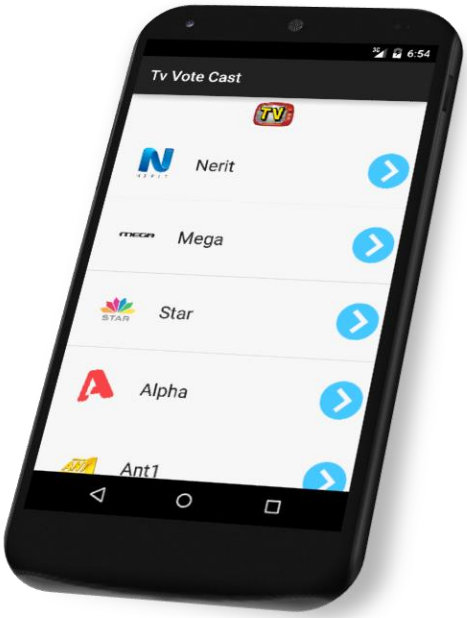
Σε επίπεδο κώδικα, τα προαναφερθέντα συμβαίνουν στο *MainActivity.java*. Το *MainActivity* ξεκινάει και αρχικοποιεί όλα τα components της παρούσας οθόνης κατά το γεγονός (event) *onCreate*. Τα components είναι: ένα listview (στο Android studio περιγράφεται ώς *channelListView*) όπου θα βρίσκεται η λίστα των καναλιών, έναν adapter (περιγράφεται ώς *mJSONAdapter*) ο οποίος αναλαμβάνει να μορφοποιήσει τα JSON δεδομένα από την κλήση στον εξυπηρετητή που τοποθετούνται στο *channelListView*, και τέλος μία καρτέλα που ζητάει τα στοιχεία του χρήστη. Η μέθοδος DisplayUserWelcome() πραγματοποιεί την τελευταία πράξη ελέγχοντας τα *shared preferences*[21] τα οποία βρίσκονται σε θέση αποθήκευσης στην συσκευή του κινητού η οποία δεν καθαρίζεται από τον συλλέκτη σκουπιδιών της Java, ούτε χάνεται μετά το κλείσιμο της εφαρμογής, ούτε ακόμα και μετά το σβήσιμο του κινητού. Τα δεδομένα χάνονται μόνο όταν γίνει απ' εγκατάσταση η εφαρμογή, δηλαδή στην περίπτωση που πλέον δεν θα είναι χρήσιμα, καταφέρνοντας έτσι να διατηρείται καθαρό το κινητό και να μην χαλάει τις βασικές λειτουργίες του λογισμικού επιβαρύνοντάς το.

Εάν λοιπόν στα *shared preferences* δεν υπάρχει εγγραφή χρήστη, δημιουργείται το AlertDialog που περιμένει τον χρήστη να καταχωρήσει το όνομά του. Ο χρήστης πατώντας το κουτί καταχώρησης κειμένου (TextView) γράφει το όνομά του, ώστε εγκρίνοντας με το κουμπί `Hi ☻` να καταχωρείται στα *shared preferences* και να γίνεται πελάτης.

Η εικόνα 12 απεικονίζει την καταχώρηση του ονόματος στα *shared preference* της εφαρμογής. Εφόσον γίνει η καταχώρηση ο χρήστης περιηγείται στην διαδικασία της επιλογής καναλιού.

Επιλογή Καναλιού

Η επιλογή καναλιού απεικονίζεται στην εικόνα 13 και βρίσκεται, όπως και ο έλεγχος για την καταχώρησης χρήστη, στο *MainActivity.java*. Η επιλογή του καναλιού γίνεται αφού η καρτέλα καταχώρησης ονόματος έχει διαγραφεί από την οθόνη και ο πελάτης, έχει δει τα διαθέσιμα κανάλια, έχει αποφασίσει ποιο θα επιλέξει και διαλέγει από ένα. Κατά την διαδικασία αυτή καλείται η μέθοδος *onItemClick()* η οποία αναλαμβάνει να φτιάξει την πρόθεση της εφαρμογής να μεταβεί στην επόμενη οθόνη, στην Java η πρόθεση αυτή ονομάζεται Intent. Στο αντικείμενο Intent τοποθετείται επίσης η μεταβλητή *channelId* που περιέχει τον μοναδικό αριθμό του επιλεγμένου καναλιού στην βάση, έτσι το επόμενο Activity θα γνωρίζει το δεδομένο αυτό. Παρακάτω, υφίσταται η επιλογή εκπομπής.



Εικόνα 13 - Επιλογή καναλιού

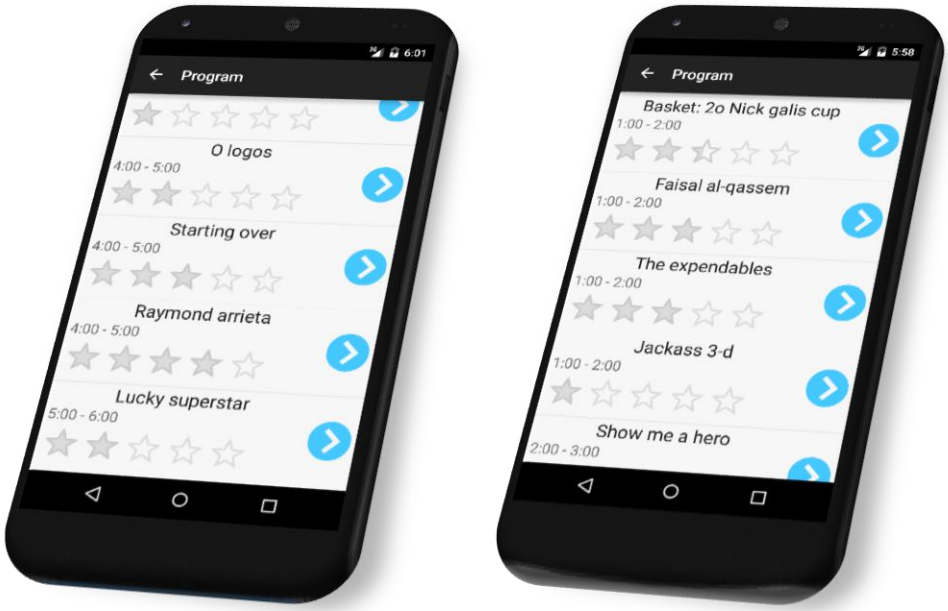
Επιλογή εκπομπής

Η επιλογή εκπομπής επεικονίζεται στην εικόνα 14. Το *ChannelDetailsActivity.java* **περιέχει τον κώδικα ανάπτυξης της λειτουργίας αυτής. Στο** *onCreate*, **όπως και στο προηγούμενο Activity δημιουργούνται όλα τα σημαντικά components της οθόνης, η εμφανιζόμενη λίστα τροφοδοτείται με δεδομένα από τον αντίστοιχο adapter (***mJSONAdapterProgram***) του καινούριου Activity.**

Ο χρήστης κοιτάει την λίστα που του εμφανίζεται και επιλέγει για ποιες από τις εκπομπές θέλει να ασκήσει την ψήφο του. Η μέθοδος *onItemClick()* **της** *programListView* **λίστας αναλαμβάνει να δημιουργήσει την πρόθεση της εφαρμογής για την επόμενη οθόνη, δηλαδή όπως και πριν, το αντικείμενο Intent. Το τελευταίο θα περιέχει επίσης τις εξής μεταβλητές που θα χρησιμοποιηθούν στην ερχόμενη οθόνη:**

1. **tvProgramId, ο μοναδικός αριθμός του προγράμματος**
2. **showName, το όνομα της εκπομπής που επιλέχτηκε**
3. **showTime, ο χρόνος παρουσίασης της εκπομπής**
4. **voteRating, η παρούσα βαθμολογία**
5. **showId, ο μοναδικός αριθμός της εκπομπής**
6. **categoryId, ο μοναδικός αριθμός της κατηγορίας**
7. **categoryName, το όνομα της κατηγορίας**

Τα δεδομένα αυτά βρίσκονται στο *programListView*, **το οποίο έχει κάνει την δικιά του αντιστοιχία της κάθε γραμμής με τα 7 στον αριθμό δεδομένα που θα αποσταλούν στο** *programVoteActivity.java* **οπού θα γίνει η κριτική της εκπομπής.**

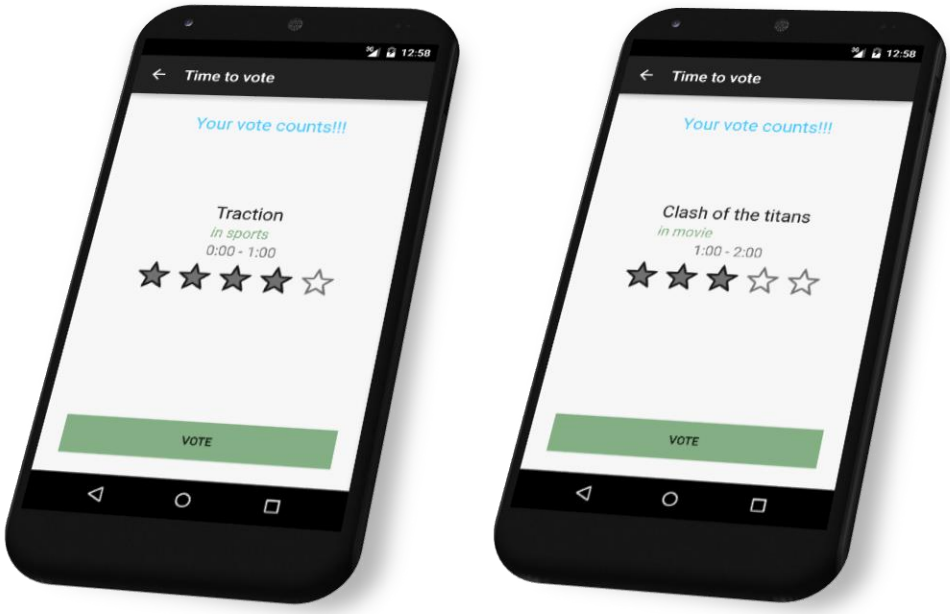


Εικόνα 14 - Επιλογή εκπομπής

Κριτική εκπομπής

Η κριτική εκπομπής απεικονίζεται στην εικόνα 15. Το *programVoteActivity.java* **περιέχει τον κώδικα ανάπτυξης της λειτουργίας αυτής. Όπως πάντα στο** *onCreate* **δημιουργούνται τα βασικά components που θα εμφανιστούν και θα χρησιμοποιηθούν κατά την γενικότερη λειτουργία της οθόνης. Η ταμπέλα (τύπου** *TextView*) **παρουσίασης του ονόματος της επιλεγμένης εκπομπής με βάση το** *showName*, **η ταμπέλα (τύπου** *TextView*) **παρουσίασης της κατηγορίας της επιλεγμένης εκπομπής με βάση το** *categoryName*, **η ταμπέλα (τύπου** *TextView*) **παρουσίασης της ώρας προβολής, πέντε**

αστέρια (τύπου *Rating Bar*) με προεπιλεγμένη τιμή την παρούσα βαθμολογία της εκπομπής, πάνω στα οποία ο χρήστης θα επιλέγει την δική του κριτική, τέλος ένα κουμπί (τύπου *Button*) που θα γράφει "Vote", είναι μερικά από τα components που αρχικοποιούνται κατά το *onCreate*.



Εικόνα 15 - Κριτική εκπομπής

Για τον χρήστη η τελική επιβεβαίωση της ψηφοφορίας γίνεται στο γεγονός του πατήματος του κουμπιού "Vote" και πιο συγκεκριμένα στην μέθοδο *btnSubmitVoteClick()*. Η μέθοδος διαβάζει το όνομα του τρέχων χρήστη από τα *shared preferences* της εφαρμογής, όπως είχε καταγραφεί κατά την είσοδο του χρήστη στην εφαρμογή, και ύστερα ωθεί τα δεδομένα που έδωσε ο χρήστης προς τον εξυπηρετητή. Τα δεδομένα που στέλνει μέσω της μεθόδου GET είναι:

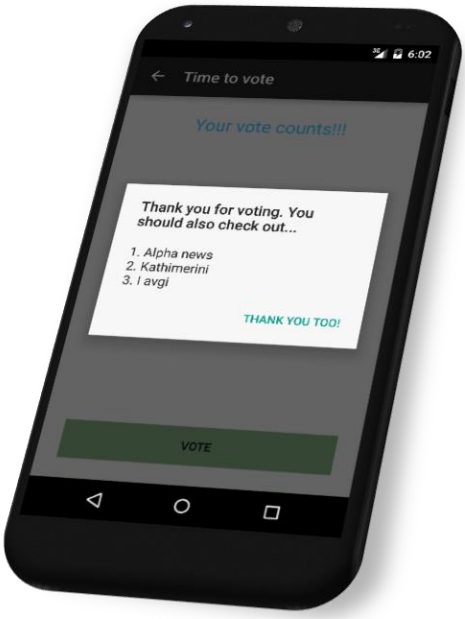
1. **tvProgramId**, ο μοναδικός αριθμός του προγράμματος
2. **voteRating**, ο αριθμός βαθμολόγισης της εκπομπής
3. **showId**, ο μοναδικός αριθμός της εκπομπής
4. **categoryId**, ο μοναδικός αριθμός της κατηγορίας
5. **userName**, το όνομα του χρήστη

Τα παραπάνω δεδομένα επεξεργάζονται από τον εξυπηρετητή, δηλαδή καταχωρούνται στην κεντρική βάση δεδομένων του Tv Vote Cast και χρησιμοποιούνται για την παραγωγή προτεινόμενων εκπομπών. Τρείς από αυτές επιστρέφονται σε μορφή JSON

στο τρέχων Activity (*programVoteActivity.java*) για να γίνει η παρουσίασή τους στην παρακάτω λειτουργία.

Παρουσίαση προτεινόμενων εκπομπών

Η παρουσίαση των προτεινόμενων εκπομπών γίνεται κατα το πέρας της επιτυχημένης ασύγχρονης κλήσης προς τον εξυπηρετητή, η κλήση γίνεται ασύγχρονα για να μην παγώνει το κινητό όσο περιμένει τον εξυπηρετητή και τον χρόνο του διαδικτύου. Στο *onSuccess* του *client* διαβάζονται οι τρείς εκπομπές "show_1", "show_2", "show_3" του JSON πίνακα "suggested_show", όπως παρουσιάζεται στην εικόνα 16



Εικόνα 16 - Κριτική εκπομπής

Η καρτέλα που εμφανίζεται έχει τίτλο "Thank you for voting. You should also check out...", στο κέντρο έχει αριθμημένες σε φθίνουσα σειρά τις προτεινόμενες εκπομπές, και κάτω δεξιά βρίσκεται το κουμπί "Thank you too!" που τερματίζει την ροή των διαδικασιών και η εφαρμογή ξεκινάει την ροή της από την αρχή.

JSON adapters

Το έργο (android project) περιέχει τα *JSONAdapter.java* **και** *JSONAdapterProgram.java* **τα οποία βοηθούν το** *ChannelDetailsActivity* **και το** *ProgramVoteActivity* **αντίστοιχα να γεμίσουν τις λίστες που παρουσιάζουν τα δεύτερα, με στοιχεία από την βάση δεδομένων. Οι Adapters επίσης, αναλαμβάνουν να σερβίρουν τα δεδομένα της κάθε γραμμής της λίστας σε όποιο αντικείμενο το θελήσει, χρησιμοποιώντας τις μεθόδους** *getItemId()*, *getView()*, *getShowName()*, *getVoteRating()* **και άλλες όπως περιγράφονται στα δύο αντικείμενα τύπου Adapter.**

String.xml

Το *string.xml* **περιέχει τα κείμενα τα οποία δεν είναι δυναμικά από την βάση δεδομένων, καθώς επίσης και την διεύθυνση του server ώστε να μπορεί εύκολα να αλλαχτεί.**

Main Layouts

Το android MVC περιέχει ένα φάκελο ο οποίος περιλαμβάνει όλα τα xml αρχεία που περιγράφουν το εικαστικό περιβάλλον. Στην εφαρμογή του Tv Vote Cast η πρώτη και δεύτερη οθόνη, περιγράφονται στο *activity_main.xml* **και** *activity_channel_details.xml* **αντίστοιχα, περιέχουν δύο listview. Το listview είναι ένα component που παρουσιάζει τα αντικείμενά του σε μορφή λίστας.**

Η οθόνη που δίνει την δυνατότητα στον χρήστη να ψηφίσει για την εκπομπή που επέλεξε περιγράφεται στο *activity_program_vote.xml*

Listview inner Layouts

Οι λίστες που παρουσιάστηκαν προηγουμένως περιέχουν ένα σύνολο από components (εικόνες, κείμενα, rating bar) που δημιουργούνται δυναμικά σύμφωνα με τα δεδομένα που λαμβάνονται από την βάση δεδομένων. Η βασική τους περιγραφή με το τί θα παρουσιάζουν είναι στα channel_list.xml και channel_list.xml

Activities

Ο κώδικας που τρέχει κατά την περιήγηση του χρήστη βρίσκεται στα Activities. *MainActivity* **για την πρώτη οθόνη που ο χρήστης επιλέγει το κανάλι,** *ChannelDetailsActivity* **για την επόμενη όπου ο χρήστης επιλέγει την εκπομπή, και τέλος** *ProgramVoteActivity* **για την τελευταία που υποβάλλεται η ψήφος.**

7.3 Αναλυτική επεξήγηση κώδικα Php

Το κομμάτι του εξυπηρετητή (Server) προγραμματίζεται σε κώδικα php. Η γενική φιλοσοφία του κώδικα αυτού είναι να δημιουργεί μία γέφυρα μεταξύ της βάσης δεδομένων και της εφαρμογής στο κινητό. Τα αρχεία που βρίσκονται στον Server και περιέχουν αυτόν τον κώδικα είναι τα *getChannelList.php*, getProgramList.php, setVote.php, recommend.php **και** *dbConnect.php.* **Το τελευταίο συμπεριλαμβάνεται στα προηγούμενα και χρησιμοποιείται για να γίνει η σύνδεση με την βάση δεδομένων ώστε να εκτελεστούν τα ανάλογα ερωτήματα. Επίσης, αρμοδιότητα των παραπάνω είναι να μετατρέπουν το αποτέλεσμα των ερωτήσεων στην βάση, σε μορφή JSON, ώστε να γίνει σωστά η επικοινωνία με το κινητό.**

dbConnect.php

Το εν λόγω αρχείο ορίζει τα στοιχεία της βάσης δεδομένων του Tv Vote Cast. Τα στοιχεία που χρειάζονται είναι η διεύθυνση της βάσης, τον χρήστη που θα συνδεθεί στην βάση, το password αυτού και το όνομα της βάσης που αντιστοιχεί στην εφαρμογή.

Για το Tv Vote Cast, η διεύθυνση βρίσκεται στην μεταβλητή *$host* **και έχει τιμή** localhost. **Ο εξυπηρετητής βρίσκεται στο ίδιο φυσικό μηχάνημα με την βάση δεδομένων, άρα η διεπαφή τους βρίσκεται στην IP διεύθυνση που και τα δύο συστήματα γνωρίζουν ως** *localhost.*

Ο χρήστης που χρησιμοποιείται βρίσκεται στην μεταβλητή *$username* **και έχει τιμή** *root*, **ο οποίος έχει πλήρη δικαιώματα ανάγνωσης και εγγραφής στην βάση.**

Ο κωδικός βρίσκεται στην μεταβλητή *$password* και έχει κενή τιμή. Αν η εφαρμογή προοριζόταν για επίπεδο παραγωγής η τιμή αυτής θα ήταν ένας περίπλοκος κωδικός.

Όλοι οι παραπάνω παράμετροι συνθέτουν την είσοδο της μεθόδου *mysqli_connect()* που πραγματοποιεί την σύνδεση με την βάση. Για την σύνδεση σε μία βάση δεδομένων MySQL, υπάρχουν τρία API (Application Programming Interface) που μπορούν να χρησιμοποιηθούν. Το καθένα έχει τα πλεονεκτήματα και μειονεκτήματά του.

- **Η MySQL επέκταση της Php**
- **Η MySQLi επέκταση της Php**
- **Php Data Objects (PDO)**

Ο κώδικας Php αποτελείται από έναν πυρήνα, με προαιρετικές επεκτάσεις(extensions) της λειτουργικότητας του πυρήνα. Οι επεκτάσεις της Php που έχουν σχέση με την βάση δεδομένων MySQL, όπως η επέκταση MySQL και MySQLi, έχουν υλοποιηθεί με την χρήση του Php framework. Μία επέκταση, συνήθως εκθέτει ένα API για τον προγραμματιστή, ώστε να μπορεί να χρησιμοποιήσει τις λειτουργίες της βάσης. Ωστόσο, ορισμένες επεκτάσεις δεν εκθέτουν API, για παράδειγμα η επέκταση MySQL PDO παρέχει μία διεπαφή στο επίπεδο PDO που βρίσκεται πάνω από την βάση.

Η επέκταση MySQLi είναι μία βελτιωμένη έκδοση του παλιότερου οδηγού MySQL, προσφέροντας διάφορα πλεονεκτήματα, από τα οποία τα πιο σημαντικά είναι τα ακόλουθα (σύμφωνα με την ιστοσελίδα της Php[22]):

- **η διεπαφή είναι αντικειμενοστραφής**
- **υποστηρίζει προετοιμασμένα αιτήματα (prepared statements)**
- **υποστηρίζει πολλαπλά αιτήματα**
- **υποστηρίζει transactions**
- **έχει βελτιωμένο debugging**
- **έχει ενσωματωμένη υποστήριξη προς τον εξυπηρετητή**
- **παρέχει πιο ισχυρή λειτουργικότητα**

Ο πίνακας 3 παρουσιάζει την σύγκριση των χαρακτηριστικών μεταξύ των τριών.

	Επέκταση MySQLi	PDO	Επέκταση MySQL
Php έκδοση που εισήχθητε	5.0	5.0	Πρίν την 3.0
Συμπεριλαμβάνεται στην έκδοση 5.χ	Ναι	Ναι	Ναι
Κατάσταση ανάπτυξης	Ενεργή ανάπτυξη	Ενεργή ανάπτυξη απο την έκδοση 5.3 και μετά	Μόνο συντήριση
Συνιστάται απο την ίδια την MySQL για καινούρια έργα	Ναι - προτινόμενη επιλογή	Ναι	Όχι
Υποστηρίζει σύνολα χαρακτήρων (Charsets)	Ναι	Ναι	Όχι
Υποστηρίζει προετημασμένα αιτήματα απο την μεριά του εξυπηρετητή	Ναι	Ναι	Όχι
Υποστηρίζει προετημασμένα αιτήματα απο την μεριά του πελάτη	Όχι	Ναι	Όχι
Υποστηρίζει αποθηκευμένες διαδικασίες (stored procedures)	Ναι	Ναι	Όχι
Υποστηρίζει πολλαπλά αιτήματα	Ναι	Τα περισσότερα	Όχι
Υποστηρίζει όλες τις λειτουργίες της MySQL 4.1+	Ναι	Τις περισσότερες	Όχι

Πίνακας 3 - σύγκριση MySQLi, PDO, MySQL

Πίσω στο αρχείο της εφαρμογής dbConnect.php, εφόσον εκτελεστεί η εντολή *mysqli_connect* **πραγματοποιείται έλεγχος για την ορθότητα της σύνδεσης. Εάν υπάρχει κάποιο λάθος στην σύνδεση με την βάση ή εάν κατά την διάρκεια της εκτέλεσης του κώδικα υπάρξει κάποιο κρίσιμο λάθος, τότε αναλαμβάνει το** *try-catch Exception*. **Στο**

catch δημιουργείται ένα αντικείμενο JSON που περιέχει το πεδίο *'success'* και το πεδίο *'result'*, με τιμές false και το μήνυμα λάθους αντίστοιχα, στην περίπτωση που δεν έγινε σύνδεση στην βάση το μήνυμα αυτό θα είναι "Failed to connect to database". Ύστερα το αντικείμενο αποστέλνεται στην κινητή συσκευή ώστε να είναι ενήμερη για την αποτυχία της διαδικασίας. Στο τέλος της διαδικασίας του catch εκτελείται η μέθοδος *exit()* η οποία τερματίζει την συνέχιση του κώδικα καθολικά.

getChannelList.php

Το αρχείο αυτό καλείται κατά την δημιουργία της λίστας των καναλιών (πρώτη οθόνη). Σαν είσοδο δεν δέχεται παραμέτρους, σαν έξοδο δίνει την λίστα των καναλιών και πιο συγκεκριμένα το *channel_id* ο μοναδικός αριθμός του καναλιού, το channel_name το όνομα του καναλιού, το *channel_icon* το όνομα της εικόνας του καναλιού.

Αναλυτικά, το *getChannelList.php* καλεί το *dbConnect.php* ώστε να συνδεθεί με την βάση δεδομένων, έτσι δημιουργείται η μεταβλητή *$conn* η οποία είναι το αντικείμενο που αντιπροσωπεύει την σύνδεση με την βάση. Ύστερα πραγματοποιείται το ερώτημα...

```
SELECT * FROM  `channel`
```

στην βάση, που επιστρέφει όλα τα στοιχεία των καναλιών, δηλαδή τα channel_id, channel_name και channel_icon. Τα στοιχεία σπρώχνονται στον πίνακα $channelArray μέσω της μεθόδου array_push() κατά την διάρκεια ενός ατέρμονου βρόγχου για όσο υπάρχουν γραμμές προς ανάγνωση. Ο τελικός πίνακας *$channelArrayResult* είναι αυτός που μετατρέπεται σε μορφή JSON και τελικά αποστέλνεται στην κινητή συσκευή του χρήστη· περιέχει τα πεδία *'success'* και *'result'*. Το πρώτο εφόσον όλα τελείωσαν σωστά είναι *true*, και το δεύτερο περιέχει το αποτέλεσμα του ερωτήματος στην βάση, δηλαδή τον πίνακα των καναλιών. Τέλος εκτελείται η εντολή αποσύνδεσης από την βάση *mysqli_close($conn)*.

Εάν κατά την διάρκεια της εκτέλεσης του κώδικα συμβεί κάποιο κρίσιμο λάθος (Exception), το *try-catch* που καλύπτει όλον τον κώδικα αναλαμβάνει να δημιουργήσει τον πίνακα *$channelArrayResult*, να τον μετατρέψει σε μορφή JSON και να στείλει απάντηση στον χρήστη. Στην περίπτωση αυτή το πεδίο *'success'* γίνεται *false* και το πεδίο *'result'* αναγράφει το μήνυμα λάθους. Επίσης, για να διασφαλιστεί η άμεση αποσύνδεση της βάσης σε παν ενδεχόμενο, εκτελείται η εντολή *mysqli_close($conn)*.

getProgramList.php

Το αρχείο αυτό καλείται κατά την δημιουργία της λίστας των προγραμμάτων (δεύτερη οθόνη). Σαν είσοδο δέχεται την ταυτότητα του καναλιού -channelId-, που έχει επιλεχθεί στο προηγούμενο μενού, σαν έξοδο δίνει την λίστα των εκπομπών και πιο συγκεκριμένα το tv_program_id, channel_id, channel_name, channel_icon, show_id, show_name, show_start, show_end, vote_rating.

Υπάρχουν δύο τρόποι που μπορεί ένα πρόγραμμα περιήγησης να στείλει πληροφορίες στον εξυπηρετητή.[23]

- **μέθοδος GET**
- **μέθοδος POST**

Πριν το πρόγραμμα περιήγησης στείλει τις πληροφορίες, τις κωδικοποιεί υπό τους όρους του σχήματος URL. Σε αυτό το σχήμα, υπάρχουν ζευγάρια ονόματος/τιμής τα οποία ενώνονται με το σύμβολο ίσον (=), και τα ζευγάρια διαχωρίζονται μεταξύ τους από το συμπλεκτικό σύμβολο (&).

Παράδειγμα:

```
name1=value1&name2=value2&name3=value3
```

Τα κενά αφαιρούνται και αντικαθυστόνται με τον χαρακτήρα +, οποιοσδήποτε άλλος μη αλφαριθμητικός χαρακτήρας αντικαθίσταται με την αντίστοιχη δεκαεξαδική τιμή. Αφού γίνει αυτή η κωδικοποίηση, τότε στέλνεται στον εξυπηρετητή.

Η μέθοδος GET στέλνεις τις κωδικοποιημένες πληροφορίες, προσηρτημένες στην αίτηση της σελίδας. Η σελίδα και η κωδικοποιημένη πληροφορία διαχωρίζονται από τον χαρακτήρα (?).

URL του Tv Vote Cast:

```
http://localhost/tv_vote_cast/getProgramList.php?channelId=1
```

Χαρακτηριστικά της μεθόδου GET

- παράγει ένα μεγάλο κείμενο χαρακτήρων το οποίο καταγράφεται στον εξυπηρετητή και είναι εμφανές στο κουτί περιήγησης του προγράμματος περιήγησης
- περιορίζεται στους 1024 χαρακτήρες
- δεν χρησιμοποιείται ποτέ για την αποστολή κωδικών ή άλλων ευαίσθητων δεδομένων
- δεν μπορεί να χρησιμοποιηθεί για την αποστολή δεδομένων σε δυαδική μορφή, όπως φωτογραφίες ή αρχεία ήχου

Η μέθοδος POST μεταφέρει τις πληροφορίες μέσω της επικεφαλίδας του HTTP. Η πληροφορία κωδικοποιείται με τον ίδιο τρόπο όπως στην μέθοδο GET και τοποθετείται στην επικεφαλίδα που ονομάζεται QUERY_STRING .

Χαρακτηριστικά της μεθόδου POST

- δεν έχει περιορισμό στο μέγεθος των δεδομένων που θα σταλούν
- μπορεί να χρησιμοποιηθεί για την αποστολή δεδομένων σε δυαδική μορφή καθώς επίσης και ASCII
- Η αποστολή γίνεται μέσω των επικεφαλίδων του HTTP, οπότε η ασφάλεια εξαρτάται από το πρωτόκολλο που χρησιμοποιείται. Με την χρησιμοποίηση του HTTPS μπορεί να διασφαλιστεί η ασφάλεια της πληροφορίας κατά ένα μεγάλο ποσοστό

Αναλυτικά ο κώδικας του *getProgramList*, πρώτα ελέγχει αν η μέθοδος GET βρίσκει την ταυτότητα του καναλιού με την οποία θα αναζητήσει στοιχεία στην βάση, αν αυτό δεν συμβεί με επιτυχία σταματάει η ροή του προγράμματος με το *throw new Exception('Wrong input')*. Στην συνέχεια αφού γίνει επίκληση του *dbConnect.php* ώστε να γίνει η σύνδεση με την βάση, πραγματοποιείται το ερώτημα...

```
SELECT * FROM `program_view` WHERE channel_id = $channelId
```

...στην βάση. Ακολουθεί μία διαδικασία επανάληψης για την κάθε γραμμή των αποτελεσμάτων που γεμίζουν τον πίνακα *$channelArray*. Ο τελευταίος αποτελεί τις πληροφορίες που απαρτίζουν το πεδίο *'result'* του τελικού πίνακα *$channelArrayResult*. Αφού προστεθεί και το πεδίο *'success'* με τιμή *true* όπως και στις προηγούμενες περιπτώσεις, τότε στέλνεται η απάντηση σε μορφή JSON. Όπως πάντα στο τέλους του προγράμματος κλείνει η σύνδεση με την βάση.

Σε περίπτωση κρίσιμου λάθους κατά την διάρκεια της εκτέλεσης του προγράμματος, ακολουθείται η ίδια διαδικασία *try-catch Exception*, **παρόμοια με αυτή του προγράμματος του αρχείου getChannelList.php.**

SetVote.php

Το setVote καλείται κατά την εκτέλεση του πατήματος του κουμπιού "Vote" στην οθόνη με τίτλο "Time to vote". Σαν είσοδο δέχεται τον μοναδικό αριθμό που αντιπροσωπεύει το πρόγραμμα που έχει επιλέξει ο χρήσης, καθώς επίσης και την αξιολόγησή του. Το αρχείο κώδικα αναλαμβάνει να ενημερώσει την βάση δεδομένων του συστήματος, αλλά και να βρει την προτεινόμενη εκπομπή που στην συνέχεια θα συμπεριλάβει μαζί με το μήνυμα επιτυχίας που θα επιστρέψει στον χρήστη. Έτσι θα εμφανιστεί ένα φιλικό μήνυμα ολοκλήρωσης της διαδικασίας στον χρήστη και τρείς εκπομπές που προτάθηκαν από το σύστημα.

Το πρόγραμμα ξεκινάει με την ανάγνωση των μεταβλητών εισόδων και τον έλεγχο αν όντως υπάρχουν. Οι μεταβλητές είναι το *vote rating* **που αντιπροσωπεύει τον δεκαδικό αριθμό που καταχώρησε ο χρήστης ως αξιολόγηση για την εκπομπή, το** *tvProgramId* **που αντιπροσωπεύει την ταυτότητα του προγράμματος στον πίνακα** *'tv_program'*, **το** *showId* **το οποίο είναι ο μοναδικός αριθμός που αντιπροσωπεύει την εκπομπη, και το** *userName* **που είναι το όνομα του χρήστη που άσκησε κριτική. Αν έστω ένα από αυτά δεν υπάρχουν τότε διεγείρεται κρίσιμο λάθος και τερματίζεται ο κώδικας επιστρέφοντας μήνυμα λάθους "Wrong input". Μόλις γίνει η σύνδεση με την βάση του Tv Vote Cast ξεκινάει το πρώτο ερώτημα σε αυτή, το οποίο είναι η ενημέρωση της εκπομπής για την κριτική του χρήστη στον πίνακα** *'show'*.

```
UPDATE `show`
SET `vote_sum` = `vote_sum`+$voteRating,
`total_votes` = `total_votes`+1
WHERE `show_id` = $showId;
```

Η στήλες που πρέπει να ενημερωθούν είναι το *vote_sum* **και το** *total_votes*. **Στην πρώτη προστίθεται το** *$voteRating* **και και προστίθεται μία ψήφος στο σύνολο των ψήφων. Το μήνυμα επιτυχίας καταχωρείται αμέσως στον πίνακα** *$queryResultArray* **στα πεδία** *'success'* **και** *'result'* **τα οποία παίρνουν τιμές** *true*.

Ακολουθεί καταγραφή στοιχείων σχετικά με των χρήστη. Πρώτα εκτελείται ερώτημα για το αν υπάρχει ο χρήστης ήδη στην βάση.

```
SELECT COUNT(`user_id`) AS 'user_count',
`user_id` AS 'user_id'
FROM `user_identity`
WHERE `user_name` = '$userName';
```

Αν το *user_count* είναι μεγαλύτερο του μηδενός σημαίνει ότι ο χρήστης υπάρχει. Αν δεν υπάρχει τότε εκτελείται εγγραφή στον πίνακα *'user_identity'* σύμφωνα με το παρακάτω.

```
INSERT INTO `user_identity` (`user_name`)
VALUES ('$userName')
```

Το *user_id* του πίνακα *'user_identity'* είναι αυτόματα αυξανόμενο, άρα δεν χρειάζεται να δηλωθεί στο ερώτημα της εισχώρησης.

Πλέον ο χρήστης, είτε υπήρχε ήδη είτε όχι είναι εγγεγραμμένος στην βάση, οπότε το πρόγραμμα προχωράει στην καταχώρηση των προτιμήσεων του χρήστη. Ο αλγόριθμος του συνεργατικού φιλτραρίσματος απαιτεί να γνωρίζει όλες τις προτιμήσεις όλων των χρηστών ξεχωριστά, και για αυτόν τον λόγο αποκλειστικά έχει δημιουργηθεί ο πίνακας *'user_preferences'*. Στον πίνακα αυτό καταγράφεται το όνομα του χρήστη (*user_id*), ο μοναδικός αριθμός που αντιπροσωπεύει την εκπομπή (*show_id*) και ο βαθμός αξιολόγησης του χρήστη (*user_rate*) για την εκάστοτε εκπομπή. Όπως αναφέρθηκε στο κεφάλαιο 5.3, κλειδί του *'user_preferences'* είναι ο συνδιασμός του *user_id* και του *show_id*, το ερώτημα που πραγματοποιεί το πρόγραμμα προς την βάση διασφαλίζει την τήρηση του κανόνα.

```
INSERT INTO `user_preference` (`user_id`, `show_id`, `user_rate`)
VALUES ($userId, $showId, '$voteRating')
ON DUPLICATE KEY UPDATE
user_rate='$voteRating'
```

Στην περίπτωση που υπάρχει το κλειδί στον πίνακα, γίνεται *UPDATE*, αλλιώς πραγματοποιείται το *INSERT*.

Σε αυτό το σημείο το πρόγραμμα έχει σκοπό να δημιουργήσει και να μορφοποιήσει τα δεδομένα για τον αλγόριθμο. Πρέπει να δημιουργηθεί ένα associative array, το οποίο είναι ένας πίνακας που έχει χαρακτήρες ως δείκτες και αποθηκεύει τα στοιχεία σε ζευγάρια κλειδιού-τιμής. Το πρόγραμμα ονομάζει αυτόν τον πίνακα *$userPreferencesArray*.

Παράδειγμα associative array για τον αλγόριθμο συνεργατικού φιλτραρίσματος για το Tv Vote Cast:

```
"Petros" =>
    Array( "24 Sport" => 2.5, "The Gladiator" => 3.5),
"Maria" =>
    Array( "The voice" => 2.5, "Hot seat" => 3.5, "World party" = > 1),
...
```

Η συσχέτιση που πραγματοποιεί ο αλγόριθμος είναι μεταξύ των χρηστών, δηλαδή βρίσκει χρήστες με παρόμοιες προτιμήσεις και προβλέπει ποια εκπομπή θα αρέσει στον τρέχων χρήστη. Το Tv Vote Cast δύναται να προτείνει εκπομπές μόνο από την ίδια κατηγορία, πρώτο μέλημα για τον σκοπό αυτό είναι να βρει την κατηγορία που ανήκει ρωτώντας την βάση.

```
SELECT `category`.`category_id` AS 'category_id'
FROM `show` LEFT JOIN `category` ON `show`.`category_id` =
`category`.`category_id`
WHERE `show_id` = $showId;
```

Το αποτέλεσμα του ερωτήματος αποθηκεύεται στην μεταβλητή *$categoryId*· θα χρησιμοποιηθεί ύστερα.

Πρώτο στοιχείο του *$userPreferencesArray* είναι τα ονόματα των υπολοίπων χρηστών της εφαρμογής, το ερώτημα που ακολουθεί επιλέγει από την βάση όλους τους χρήστες.

```
SELECT `user_name` AS 'user_name'
FROM `user_identity`
```

Ακολουθεί μία δομή επανάληψης για κάθε χρήστη, κατά την οποία δημιουργούνται οι πρώτοι δείκτες του *$userPreferencesArray* με τιμή ένα κενό, προς το παρόν, πίνακα.

Στην συνέχεια θα δημιουργηθούν οι πίνακες με τα δεδομένα των προτιμήσεων για τον κάθε χρήστη. Οπότε εκτελείται βρόγχος επανάληψης κατά τον οποίο εκτελείται το ανάλογο ερώτημα στην βάση δεδομένων από τους πίνακες *'show'*, *'user_preferences'*, *'user_identity'* και *'category'*.

```
SELECT `show`.`show_id`, `show`.`show_name`,`user_rate`
FROM `user_preference`
    LEFT JOIN `show` ON `user_preference`.`show_id` = `show`.`show_id`
    LEFT JOIN `user_identity` ON `user_preference`.`user_id` =
`user_identity`.`user_id`
    LEFT JOIN `category` ON `show`.`category_id` =
`category`.`category_id`
WHERE `user_name` = '$key' AND `category`.`category_id` = $categoryId
```

Τα αποτελέσματα αυτού καταγράφονται στην μεταβλητή *$statsArray*, η οποία στο τέλος των επαναλήψεων συγχωνεύεται στον πίνακα *$userPreferencesArray*.

Έχοντας πλέον αναζητήσει, δημιουργήσει και μορφοποιήσει τα δεδομένα, έρχεται η σειρά να εκτελεστεί ο αλγόριθμος συνεργατικού φιλτραρίσματος. Ο κώδικας υλοποίησης αυτού βρίσκεται σε ξεχωριστό αρχείο, το οποίο επικαλείται με την εντολή *require_once("recomend.php")*. Από το recommend.php καλείται η μέθοδος *getRecommendations()* η οποία δέχεται δύο ορίσματα, τον πίνακα με τις προτιμήσεις των χρηστών (*$userPreferences*) και τον τρέχων χρήστη (*$userName*). Τα αποτέλεσμα του αλγορίθμου αποθηκεύεται στην μεταβλητή *$suggested_show*, από τα οποία τα τρία πρώτα εισχωρούντε στο *$queryResultArray['suggested_show']* στα πεδία *show_1*, *show_2* και *show_3* αντίστοιχα. Τέλος, όπως σε όλες τις διαδικασίες του εξυπηρετητή, το αποτέλεσμα μετατρέπεται σε μορφή JSON και επιστρέφεται στον πελάτη.

Όπως σε όλα τα προηγούμενα έτσι και στον *setVote*, ο κώδικας περιλαμβάνεται σε *try-catch*, οπότε αν γίνει κρίσιμο λάθος κατά την διάρκεια της εκτέλεσης, το *catch* αναλαμβάνει να δημιουργήσει το αντίστοιχο μήνυμα λάθους προς την εφαρμογή της κινητής συσκευής.

8. ΕΠΙΛΟΓΟΣ

Το έργο του Tv vote cast αναπτύχθηκε με σκοπό να γίνουν οικίες σε μένα, τον συγγραφέα και εκτελεστή της εργασίας, οι καινούριες τεχνολογίες κυρίως του Android. Για να δημιουργηθεί ένα τέτοιο έργο για εμπορικούς σκοπούς, πρέπει να συνυπάρξουν πολλά συστήματα, android application, server, βάση δεδομένων. Η υλοποίηση της εργασίας είναι system to system, που σημαίνει ότι όλα τα παραπάνω έχουν υλοποιηθεί και για αυτό το αποτέλεσμα είναι ορατό, λειτουργεί στην πράξη και όχι μόνο στην θεωρία. Δεδομένου ότι η εργασία είναι ατομική, ακολουθήθηκε το μοντέλο καταρράχτη (waterfall). Ξεκίνησα με την σχεδίαση της βάσης δεδομένων, ύστερα ανέπτυξα την εφαρμογή Android ακολουθώντας οδηγούς εκμάθησης[24] και συμβουλές που έβρισκα στο διαδίκτυο.

Η γενική ιδέα και ο σκοπός της εφαρμογής δημιουργήθηκε με γνώμονα να υπάρξει κάποιος λόγος δημιουργίας μίας τέτοιας εφαρμογής. Προσωπικά, δεν πιστεύω ότι μία τέτοια εφαρμογή θα είχε αρκετή αποδοχή από τους χρήστες, επειδή ίσως οι χρήστες δεν έχουν κέρδος, δεν προσφέρει κάποια διασκέδαση ούτε προσφέρει κάποια ουσιαστική υπηρεσία. Προσφέρει μόνο μία συμβουλή από μία τεχνητή νοημοσύνη, που δύσκολα θα επηρεάσει κάποιον αν δεν έχει πειστεί ότι είναι άξια εμπιστοσύνης. Σε ένα ιδανικό κόσμο που οι άνθρωποι ήταν τόσο καλόβουλοι και έχουν τον χρόνο να ασχοληθούν με κάτι τέτοιο, τότε θα λέγαμε ότι θα είχε κάποιο χρόνο ζωής. Θα ήταν προτιμότερο να είναι ένα κομμάτι ενός μεγαλύτερου έργου, που πέρα από τα υπόλοιπα θα παρέχει και αυτήν την υπηρεσία.

Τέλος, θα πρέπει να αναφερθεί πως ο σκελετός του Tv Vote Cast μπορεί να χρησιμοποιηθεί για άλλες ιδέες, για παράδειγμα θα μπορούσε να είναι η διαδικασία υποβολής ψηφοφορίας μίας εκπομπής στην τηλεόραση. Γενικά, το Tv Vote Cast είναι η ακτινογραφία μίας εφαρμογής που δίνει την δυνατότητα στους ανθρώπους να εκφράσουν την ικανοποίηση τους ή την δυσαρέσκεια τους για ένα προϊόν.

Πηγές

[1] Android SDK tools
http://developer.android.com/sdk/index.html

[2] IntelliJ IDEA 15.0 Help
https://www.jetbrains.com/idea/help/quick-start-guide.html

[3] What is Java technology and why do I need it?
http://java.com/en/download/faq/whatis_java.xml

[4] Java Virtual Machine Technology
https://docs.oracle.com/javase/8/docs/technotes/guides/vm/

[5] Java Garbage Collection Basics
http://www.oracle.com/webfolder/technetwork/tutorials/obe/java/gc01/index.html

[6] Just in Time Compilation
http://www.cs.columbia.edu/~aho/cs6998/Lectures/14-09-22_Croce_JIT.pdf

[7] Java SE HotSpot at a Glance
http://www.oracle.com/technetwork/articles/javase/index-jsp-136373.html

[8] Class object
https://docs.oracle.com/javase/7/docs/api/java/lang/Object.html

[9] Understanding JIT Compilation and Optimizations
https://docs.oracle.com/cd/E13150_01/jrockit_jvm/jrockit/geninfo/diagnos/underst_jit.html

[10] Java performance
https://en.wikipedia.org/wiki/Java_performance

[11] JSON
https://en.wikipedia.org/wiki/JSO

[12] Donari, D., Ordinez, L., Santos, R. (2008)
Real-Time Server Oriented Operating System for Embedded Applications

[13] Piateski, Gregory, and William Frawley.
Knowledge discovery in databases. MIT press, 1991.

[14] Db-Engines Ranking
http://db-engines.com/en/ranking

[15] Friends, Apache. "XAMPP Apache+ MySQL+ PHP+ Perl."
Disponible: https://www. apachefriends. org/es/index. html.[Último acceso: marzo 2015] (2014).

[16] Foundations and TrendsR in Human–Computer Interaction
Vol. 4, No. 2 (2010) 81–173 c 2011 M. D. Ekstrand, J. T. Riedl and J. A. Konstan

[17] F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, eds.
Recommender Systems Handbook. Springer, 2010.

[18] J. Reilly, J. Zhang, L. McGinty, P. Pu, and B. Smyth,
"Evaluating compound critiquing recommenders: A real-user study," in ACM EC '07, pp. 114–123, ACM, ACM ID: 1250929, 2007.

[19] P. Zigoris and Y. Zhang,
"Bayesian adaptive user profiling with explicit & implicit feedback," in ACM CIKM '06, pp. 397–404, ACM, 2006.

[20] Standard flowchart and their usage
https://www.edrawsoft.com/flowchart-symbols.php

[21] Storage options
http://developer.android.com/guide/topics/data/data-storage.html

[22] MySqli Overview
http://php.net/manual/en/mysqli.overview.php

[23] PHP - GET & POST Methods
http://www.tutorialspoint.com/php/php_get_post.htm

[24] Android Tutorial for Beginners, από τον Darryl Bayliss, 10 Νοεμβρίου 2014
http://www.raywenderlich.com/78574/android-tutorial-for-beginners-part-1