ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ
ΕΡΓΑΣΤΗΡΙΟ ΥΠΟΛΟΓΙΣΤΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

Οπτικοποίηση ιατρικών απεικονιστικών δεδομένων στον τρισδιάστατο χώρο στα πλαίσια ιατρικών τηλεδιαβουλεύσεων στον φυλλομετρητή ιστού

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Παύλος Π. Φραγκογιάννης

Επιβλέπων : Παναγιώτης Δ. Τσανάκας
Καθηγητής Ε.Μ.Π.

Αθήνα, Φεβρουάριος 2019

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ
ΕΡΓΑΣΤΗΡΙΟ ΥΠΟΛΟΓΙΣΤΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

Οπτικοποίηση ιατρικών απεικονιστικών δεδομένων στον τρισδιάστατο χώρο στα πλαίσια ιατρικών τηλεδιαβουλεύσεων στον φυλλομετρητή ιστού

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Παύλος Π. Φραγκογιάννης

Επιβλέπων : Παναγιώτης Δ. Τσανάκας
Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 25η Φεβρουαρίου 2019.

.....	.....	.....
Παναγιώτης Τσανάκας	Κιαμάλ Πεκμετζή	Δημήτριος Κουτσούρης
Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.

Αθήνα, Φεβρουάριος 2019

.....

Παύλος Π. Φραγκογιάννης

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Περίληψη

Η σταδιακή μεταστροφή που παρατηρείται στη σύγχρονη ιατρική επιστήμη, για τη συγκρότηση ιατρών σε πολυεπιστημονικές ομάδες που διαχειρίζονται από κοινού ιατρικά περιστατικά, εγείρει την ανάγκη για τη δημιουργία εργαλείων που θα επιτρέπουν τη συνεργασία σε πραγματικό χρόνο (online) γεωγραφικά κατανεμημένου ιατρικού προσωπικού. Έχουν πραγματοποιηθεί πολλές προσπάθειες για δημιουργία τέτοιων εργαλείων, όμως η ανάγκη για συνεχή βελτίωση των υπηρεσιών τους δεν παύει να υφίσταται. Η παρούσα διπλωματική εργασία ενισχύει μια υπάρχουσα πλατφόρμα επεξεργασίας ιατρικών δεδομένων σε πραγματικό χρόνο και ιατρικών τηλεδιασκέψεων με χαρακτηριστικά προηγμένης πολυεπίπεδης ανασύνθεσης και απεικόνισης τρισδιάστων μοντέλων, στα πρότυπα των κορυφαίων λογισμικών ιατρικής επεξεργασίας στην αγορά. Παράλληλα, αφομοιώνει την προστιθέμενη λειτουργικότητα στη λογική της πλατφόρμας για την παράλληλη επεξεργασία και συγχρονισμό των δράσεων μεταξύ των συμμετεχόντων. Η παρουσιαζόμενη εργασία αξιοποιεί τελευταίες διαδικτυακές τεχνολογίες και διεπαφές προγραμματισμού εφαρμογών (APIs) για να υποστηρίξει την επεξεργασία των ιατρικών δεδομένων από την πλευρά του ιατρού χρήστη, καθώς και το συγχρονισμό των αντικειμένων πάνω από διαύλους του WebRTC.

Λέξεις κλειδιά

τηλεϊατρική, συνεργατική ροή εργασίας, συνεργατική διάγνωση, ιατρική απεικόνιση, πρότυπο DICOM, τρισδιάστατη απεικόνιση, πολυεπίπεδη ανασύνθεση, WebGL, WebRTC, React, Redux, ami.js, vtk.js

Abstract

The gradual shift in modern medical practice, from working alone clinical doctors to MDTs (MultiDisciplinary Teams), raises the need of online real-time collaboration among geographically distributed medical personnel. There have been done many attempts toward this direction, though the need for constantly improving their services does not cease to exist. This diploma thesis strengthens an existing real-time medical data processing and teleconference platform with advanced MPR (MultiPlanar Reconstruction) and Volume Rendering features, in the standards of leading medical imaging software on the market. Moreover, it integrates the added functionality into the platform's logic for parallel processing and synchronization of actions between participants. The presented work leverages state-of-the-art features of the web (technologies and APIs) to support client-side medical data processing, as well as synchronization of actions over the WebRTC channels.

Keywords

telemedicine, collaborative workflow, collaborative diagnosis, medical imaging, DICOM standard, volume rendering, multiplanar reconstruction, WebGL, WebRTC, React, Redux, ami.js, vtk.js

Ευχαριστίες

Θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή κ. Παναγιώτη Τσανάκα για τις πολύτιμες συμβουλές και την παρότρυνση πάνω στο συγκεκριμένο αντικείμενο, καθώς και την εμπιστοσύνη που έδειξε προς το πρόσωπό μου αναθέτοντάς μου την παρούσα διπλωματική εργασία. Ακόμα, τις θερμές μου ευχαριστίες στον Χρήστο Ανδρίκο και ιδιαίτερα στον Γιώργο Ρασσιά που με την εμπειρία, την καθοδήγηση και την υπομονή τους συνέβαλαν τα μέγιστα στην εκπόνησή της. Τέλος, θα ήθελα να ευχαριστήσω την οικογένειά μου και τους φίλους μου για τη συμπαράσταση και την υπομονή τους καθ' όλη τη διάρκεια των σπουδών μου στο Ε.Μ.Π.

Περιεχόμενα

Εικόνες

Πίνακες

1 Εισαγωγή

Η σύγχρονη τάση στην παροχή υπηρεσιών υγείας είναι η χρησιμοποίηση ιατρικών πολυεπιστημονικών ομάδων (MDT - MultiDisciplinary Teams) για τη βελτίωση του ποσοστού επιτυχίας της θεραπευτικής αγωγής και την ισχυροποίηση της προληπτικής ιατρικής, αξιοποιώντας παράλληλα ιατρικούς πόρους, υλικούς και άυλους, αποδοτικότερα. Καθότι απαιτούνται δύσκολες και εξειδικευμένες δράσεις για την αντιμετώπιση περίπλοκων ιατρικών περιστατικών, οι ιατροί σήμερα αναγνωρίζουν πως η συμμετοχή περισσότερων ειδικοτήτων στη φροντίδα των ασθενών αυξάνει τις πιθανότητες επιτυχούς ανάρρωσης των τελευταίων.

Ωστόσο, η πολυάσχολη καθημερινότητα των επιστημόνων υγείας δεν ευνοεί την οργάνωση τακτικών συναντήσεων και διασκέψεων προκειμένου να διεξάγονται συλλογικές διαγνώσεις και ανταλλαγή απόψεων. Παράλληλα, συχνά η διαφορετική γεωγραφική κατανομή των επιστημόνων δεν επιτρέπει την ταυτόχρονη αλληλεπίδραση πολλαπλών επιστημόνων με τα αποτελέσματα ιατρικών εξετάσεων και την από κοινού επεξεργασία τους.

Επόμενο λοιπόν είναι η αξιοποίηση της υπάρχουσας τεχνολογίας για την ανάπτυξη εφαρμογών με σκοπό να εξαλειφθεί το πρόβλημα της γεωγραφικής απόστασης, μέσω διαβουλεύσεων σχετικά με τους ασθενείς και τις εξετάσεις αυτών. Έμφαση δίνεται στην ανάγκη οι συμμετέχοντες διαβουλεύσεων που αλληλεπιδρούν να είναι σε θέση να γνωρίζουν τι κάνουν οι υπόλοιποι όπως και να έχουν πρόσβαση στα ευρήματα μέσα στον κοινό χώρο εργασίας. Οι ανάγκες αυτές πρέπει καθοριστικά να υποστηρίζονται από ένα τέτοιο συνεργατικό σύστημα [1].

Εντούτοις, εξίσου σημαντική είναι η μεγιστοποίηση των παρεχόμενων υπηρεσιών προς τους επιστήμονες μέσω των παρεχόμενων σε αυτούς εργαλεία, προκειμένου να επιτυγχάνεται απρόσκοπτα η επιτυχής και ακριβής διάγνωση. Σε ένα τεχνολογικό κόσμο που εξελίσσεται με γοργούς ρυθμούς συχνά παραλείπεται η μεγιστοποίηση των δυνατοτήτων των υπαρχόντων υπηρεσιών στο βωμό της εξέλιξης νέων αντικειμένων και ιδεών.

1.1 Σκοπός της Διπλωματικής

Σκοπός λοιπόν της παρούσας διπλωματικής εργασίας είναι η ενίσχυση της πλατφόρμας «HERMES» του GRNET, μιας δικτυακής πλατφόρμας ιατρικών διαβουλεύσεων σε ιατρικές εικόνες, με δυνατότητες προηγμένης ιατρικής επεξεργασίας και συγκεκριμένα

πολυεπίπεδης ανασύνθεσης και απεικόνισης τρισδιάστατων μοντέλων, σε πραγματικό χρόνο.

Για την υλοποίηση αυτή ακολουθήθηκαν πιστά οι σχεδιαστικές αρχές του προγράμματος και των χρησιμοποιούμενων τεχνολογιών, κάνοντας παράλληλα τις απαραίτητες προσθήκες σε δικτυακές τεχνολογίες και APIs. Έτσι, συνεχίζοντας τη λογική του BYOD (Bring Your Own Device) που ακολουθεί η πλατφόρμα ως εφαρμογή περιβάλλοντος browser άνευ εγκατάστασης, το αποτέλεσμα είναι ένα “φορητό” συνεργατικό εργαλείο ιατρικών διαβουλεύσεων με εφάμιλλη λειτουργικότητα και πλειονότητα επιλογών των κυριότερων standalone DICOM viewers της αγοράς.

1.2 Οργάνωση κειμένου

Στο Κεφάλαιο 2 παρατίθεται μια σύντομη εισαγωγή στα προτεινόμενα πρότυπα του χώρου της υγείας, μερικά εκ των οποίων έχουν άμεση σχέση με το αντικείμενο της διπλωματικής, παρουσιάζονται κάποιες εφαρμογές ιατρικής απεικόνισης και παρουσιάζεται σχετική έρευνα που έχει γίνει στο χώρο τρισδιάστατης επεξεργασίας ιατρικών δεδομένων στη ραδιολογία. Επίσης, παρουσιάζονται κάποιες κεντρικές έννοιες του προτύπου DICOM και της ιατρικής απεικόνισης που βρίσκονται στο επίκεντρο της υλοποίησης. Στο Κεφάλαιο 3 επιχειρείται μια προσέγγιση των ποικίλων τεχνολογιών ιστού που χρησιμοποιήθηκαν για την εξέλιξη της πλατφόρμας, ενώ παράλληλα τεκμηριώνεται η σημασία τους και ο λόγος για τον οποίο επιλέχθηκαν. Το Κεφάλαιο 4 περιέχει την ανάλυση και το σχεδιασμό των δύο προς προσθήκη λειτουργιών/προγραμμάτων ως ανεξάρτητες οντότητες/προγράμματα, καθώς και μία σύντομη αναφορά επιδόσεων. Το Κεφάλαιο 5 ασχολείται με την διαδικασία που ακολουθήθηκε για την επιτυχή αφομοίωση αυτών στην πλατφόρμα και τη νέα όψη αυτής. Στο Κεφάλαιο 6 παρατίθεται μια αποτίμηση της συνολικής εργασίας καθώς επίσης και κάποιες ιδέες που μπορούν να οδηγήσουν σε μελλοντικές επεκτάσεις της πλατφόρμας. Τέλος, το Κεφάλαιο 7 αποτελείται από τη βιβλιογραφία που χρησιμοποιήθηκε κατά τη συγγραφή.

2 Σχετικά πρότυπα, εφαρμογές και Ερευνητικό υπόβαθρο

2.1 Πρότυπα ανταλλαγής ιατρικών δεδομένων

Ένα από τα ζητούμενα που υπάρχουν σήμερα στον τομέα της Υγείας είναι η ανάγκη αποτελεσματικής επεξεργασίας του τεράστιου όγκου δεδομένων (πληροφοριών) που παράγονται καθημερινά από τους διάφορους φορείς που εξυπηρετούν τον τομέα παροχής Υγείας, όπως είναι τα νοσοκομεία, τα εργαστήρια, τα ιδιωτικά θεραπευτήρια, οι ασφαλιστικοί οργανισμοί κλπ.

Όμως, ακόμη και σήμερα, και παρά τη ραγδαία εξέλιξη της επιστήμης της Πληροφορικής και της τεχνολογίας των υπολογιστών και των τηλεπικοινωνιών, τα συλλεγόμενα δεδομένα τις περισσότερες φορές δεν δέχονται ηλεκτρονική επεξεργασία. Επιπλέον, ο τομέας Πληροφορικής στον χώρο της Υγείας εξακολουθεί να αποτελείται κυρίως από αυτόνομες μονάδες με μικρή, εάν όχι ελάχιστη, ανταλλαγή δεδομένων και πληροφοριών μεταξύ τους.

Η ραγδαία εξέλιξη της τεχνολογίας έχει πλέον καταστήσει σαφές ότι σήμερα είναι τεχνολογικά εφικτή η δημιουργία ενός ανθρωποκεντρικού συστήματος παροχής υπηρεσιών υγείας. Σε ένα τέτοιο περιβάλλον θα πρέπει να υπάρχει απρόσκοπτη ανταλλαγή πληροφοριών μεταξύ όλων των προσώπων και φορέων που εμπλέκονται στην παροχή υπηρεσιών υγείας. Το σχήμα αυτό περιλαμβάνει ενημέρωση και την επικοινωνία νοσοκομείων, ιδιωτικών θεραπευτηρίων, ιδιωτών ιατρών, ασφαλιστικών φορέων, αλλά και του ίδιου του ασθενούς. Είναι φανερό ότι η λειτουργία ενός τέτοιου ολοκληρωμένου συστήματος παροχής φροντίδας υγείας προϋποθέτει τη διασυνδεσιμότητα και τη δυνατότητα ανταλλαγής πληροφοριών εννοιολογικά αναγνωρίσιμων.

Ο τομέας, λοιπόν, των επαγγελματιών υγείας, προκειμένου να επιλύσει αυτά τα θέματα, έχει ερευνήσει μεθόδους επίτευξης διαλειτουργικότητας για χρόνια. Γεγονός είναι πως αυτές οι προσεγγίσεις απέφεραν αποτελέσματα, κυρίως με τη μορφή προτασσόμενων προτύπων. Τα κυριότερα από αυτά θα εξεταστούν στη συνέχεια ενώ κάποια από αυτά θα χρησιμοποιηθούν και στα πλαίσια της παρούσας διπλωματικής εργασίας.

2.1.1 Το πρότυπο HL7

Το πρότυπο HL7 το οποίο έχει αναπτυχθεί από τον ομώνυμο οργανισμό [2] είναι το πλέον ώριμο και ευρέως χρησιμοποιημένο πρότυπο ανταλλαγής πληροφοριών μέσω μηνυμάτων στο χώρο της υγείας. Η έρευνα τόσο από την ακαδημαϊκή κοινότητα όσο και από την βιομηχανία και τις εταιρίες συμβούλων οδήγησε σ/ αυτό το πρότυπο το οποίο μπορεί πράγματι να χρησιμοποιηθεί στην πράξη. Η κυριότητά του ανήκει στο μη

κερδοσκοπικό οργανισμό Health Level 7 και έχει αναγνωριστεί από πολλά εθνικά ιδρύματα προτυποποίησης όπως ο ANSI (Η.Π.Α.) και ο DIN (Γερμανία).

Το HL7 είναι ένα πρωτόκολλο επικοινωνίας που μπορεί να εφαρμοστεί τόσο σε νοσοκομεία και εργαστήρια όσο και σε μονάδες διοίκησης, διαχείρισης και management των υπηρεσιών υγείας κάθε χώρας. Αυτό το πετυχαίνει διότι εξασφαλίζει την ηλεκτρονική επικοινωνία ετερογενών πληροφοριακών συστημάτων ανταλλάσσοντας δεδομένα (μέσω HL7 μηνυμάτων). Τα πληροφοριακά αυτά συστήματα μπορεί να υποστηρίζουν διαφορετικές λειτουργικές μονάδες ενός οργανισμού υγείας ή ακόμη και να ανήκουν σε διαφορετικούς οργανισμούς υγείας. Το πρότυπο HL7 λοιπόν είναι ένας κοινά αποδεκτός από όλους τους κατασκευαστές κώδικας επικοινωνίας.

Με τη χρήση του προτύπου, για παράδειγμα, μπορεί ένας αναλυτής σε εργαστήριο νοσοκομείου να δέχεται απευθείας εντολές εξετάσεων από τα κλινικά τμήματα και να επιστρέφει τις απαντήσεις των εξετάσεων που διενεργεί στα τμήματα που τις παρήγγειλαν αυτόματα. Έτσι υποβοηθιέται σημαντικά ένας κλινικός γιατρός αφού απαλλάσσεται από το φόρτο της χειρωνακτικής διαχείρισης τεράστιου όγκου ιατρικής πληροφορίας, που απορροφά σημαντικό χρόνο και τον αποσπά από τον πρωταρχικό σκοπό του, τη διάγνωση και θεραπεία του ασθενή του.

Το πρότυπο HL7 δεν αφορά αποκλειστικά τη διαβίβαση πληροφορίας μεταξύ εργαστηρίου και κλινικής. Είναι έτσι δομημένο που εκτός από κλινικά και εργαστηριακά δεδομένα εμπεριέχει και όλες τις υπαρκτές πληροφορίες σε κάθε μονάδα υγείας δηλαδή ασφαλιστικά και οικονομικά στοιχεία, προμήθειες και διαχείριση υλικών, φαρμάκων και εργαλείων, αναλώσιμων και πάγιου εξοπλισμού.

Το μόνο που απαιτείται είναι η φυσική διασύνδεση των συστημάτων και το κάθε τμήμα μπορεί να έχει τα στοιχεία που του είναι απαραίτητα για τη λειτουργία του. Έτσι διεκπεραιώνεται αυτόματα το υπόλοιπο πλην του κλινικού έργο και αποφεύγεται εντελώς η γραφειοκρατία εφόσον μία και μοναδική εγγραφή για κάθε ασθενή μπορεί να διανέμεται εύκολα και κατάλληλα σε κάθε τμήμα, κλινικό, εργαστηριακό ή διοικητικό ανάλογα με τις ανάγκες του τμήματος. Το ίδιο εύκολη είναι και η διαδικασία της ενημέρωσης της κάθε εγγραφής αφού αρκεί να γίνει αυτή η διαδικασία από ένα μόνο τμήμα. Γι' αυτό και το πρότυπο HL7 επικρατεί αφού αντιμετωπίζει ένα νοσοκομείο ή ένα ευρύτερο σύστημα, σαν ενιαία λειτουργική οντότητα, όπως πράγματι είναι.

Το πρότυπο Hl7 μπορεί να εγκατασταθεί και να λειτουργήσει στα ήδη υπάρχοντα πληροφορικά συστήματα, και στον ήδη υπάρχοντα ιατροτεχνολογικό εξοπλισμό. Δεν απαιτεί καμία αλλαγή και διασυνδέει τα συστήματα και τα μηχανήματα κάθε κατασκευαστή. Ό,τι είναι ήδη εγκατεστημένο σε ένα νοσοκομείο ή μια μονάδα υγείας, από πλευράς τεχνολογικού εξοπλισμού, κάθε είδους, με την χρήση του προτύπου HL7 συνδέεται και με τον ολόκληρο το υπόλοιπο εξοπλισμό.

2.1.2 EHR/EMR

Ως EHR (Electronic Health Record), ή EMR (Electronic Medical Record), ορίζεται μια συστηματική συλλογή από ηλεκτρονικές πληροφορίες υγείας σχετιζόμενες με ένα συγκεκριμένο ασθενή ή πληθυσμό. Είναι μια καταχώρηση σε ψηφιακή μορφή που θεωρητικά είναι σε θέση να διαμοιραστεί μεταξύ διαφορετικών φορέων υγείας. Σε μερικές περιπτώσεις, αυτός ο διαμοιρασμός υλοποιείται διαμέσου δικτυακά διασυνδεδεμένων πληροφοριακών συστημάτων σε επιχειρησιακό επίπεδο. Τα EHRs μπορεί να περιλαμβάνουν ένα μεγάλο εύρος από δεδομένα, συμπεριλαμβανομένων δημογραφικών στοιχείων, ιατρικού ιστορικού, φαρμακευτικής αγωγής και αλλεργιών, εργαστηριακών εξετάσεων, ιατρικών εικόνων, ζωτικών δεικτών, προσωπικών στοιχείων όπως ηλικία και βάρος και πληροφοριών χρέωσης παρεχόμενων ιατρικών υπηρεσιών.

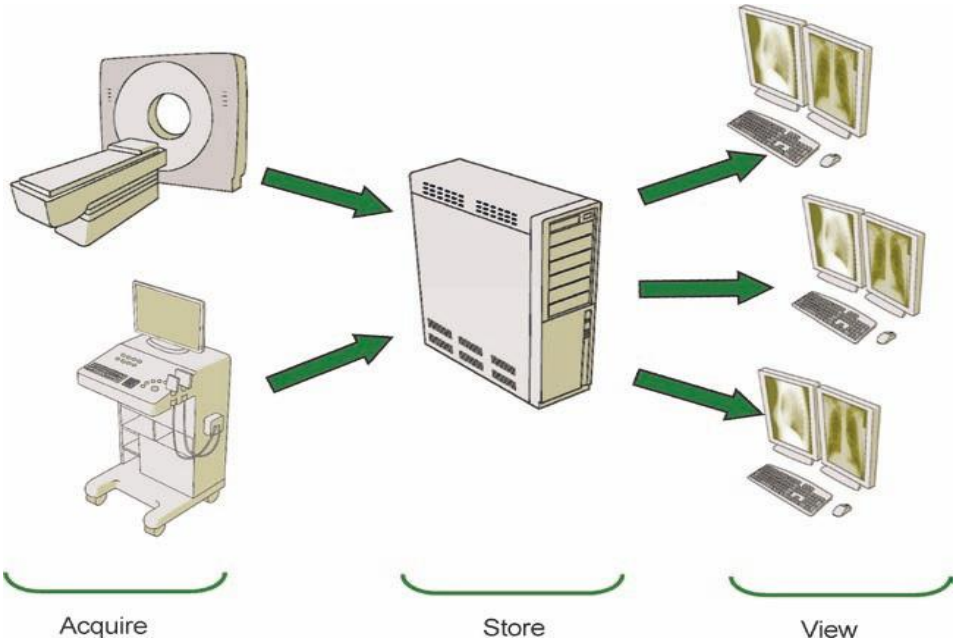
Το όλο σύστημα είναι σχεδιασμένο να αναπαριστά δεδομένα που περιγράφουν με ακρίβεια την κατάσταση του ασθενούς την εκάστοτε χρονική στιγμή. Επιτρέπει την προβολή ολόκληρου του ιστορικού του ασθενούς χωρίς την ανάγκη για εντοπισμό προηγούμενων καταχωρήσεων και εξασφαλίζει πως τα εν λόγω δεδομένα είναι πάντα ενημερωμένα και έγκυρα. Μειώνει δε την πιθανότητα των διπλοτύπων, καθώς υπάρχει μία και μοναδική καταχώρηση ανά ασθενή. Για τον τελευταίο αυτό λόγο καθιστά επίσης την εξαγωγή χρήσιμης ιατρικής πληροφορίας περισσότερο αποδοτική.

2.1.3 Το πρότυπο DICOM και τα PACS

Το DICOM αντιπροσωπεύει την ψηφιακή απεικόνιση και την επικοινωνία στην ιατρική και αντιπροσωπεύει τα χρόνια προσπάθειας για τη δημιουργία του πιο καθολικού και θεμελιώδους προτύπου στην ψηφιακή ιατρική απεικόνιση. Ως εκ τούτου, παρέχει όλα τα απαραίτητα εργαλεία για τη διαγνωστική ακριβή αναπαράσταση και επεξεργασία των δεδομένων της ιατρικής απεικόνισης. Επιπλέον, αντίθετα με τη δημοφιλή πεποίθηση, το DICOM δεν είναι απλά μια εικόνα ή ένας τύπος αρχείου εικόνας. Πρόκειται για ένα ολοκληρωμένο πρωτόκολλο μεταφοράς, αποθήκευσης και προβολής δεδομένων κατασκευασμένο και σχεδιασμένο για να καλύπτει όλες τις λειτουργικές πτυχές της ψηφιακής ιατρικής απεικόνισης (και γι’ αυτό πολλοί θεωρούν το DICOM ως ένα σύνολο προτύπων, αντί για ένα μόνο πρότυπο). Χωρίς αμφιβολία, το DICOM κυβερνά την πρακτική ψηφιακή ιατρική.

Ένα άλλο σημαντικό ακρωνύμιο που φαίνεται ότι όλες οι εταιρείες DICOM χρησιμοποιούν είναι το PACS (Αρχειοθέτηση Εικόνων και Συστήματα Επικοινωνίας). Τα PACS είναι ιατρικά συστήματα (που αποτελούνται από το απαραίτητο υλικό και λογισμικό) που έχουν σχεδιαστεί και χρησιμοποιούνται για την εκτέλεση ψηφιακής ιατρικής απεικόνισης. Περιλαμβάνουν συσκευές ανάκτησης ψηφιακής εικόνας (μηχανήματα όπως σαρωτές υπολογιστικής τομογραφίας (CT) ή υπερήχων), αρχεία ψηφιακών εικόνων (όπου

αποθηκεύονται οι εικόνες που έχουν ληφθεί) και σταθμούς εργασίας (όπου οι ακτινολόγοι βλέπουν τις εικόνες). Όταν κάποιος χρησιμοποιεί την ψηφιακή του φωτογραφική μηχανή, αποθηκεύει τις εικόνες στον υπολογιστή του και τις στέλνει στους φίλους του, χρησιμοποιεί το ίδιο ακριβώς μοντέλο. Φυσικά, το μοντέλο του PACS το πηγαίνει σε ένα πολύ πιο σύνθετο επίπεδο (Εικόνα 2-1).



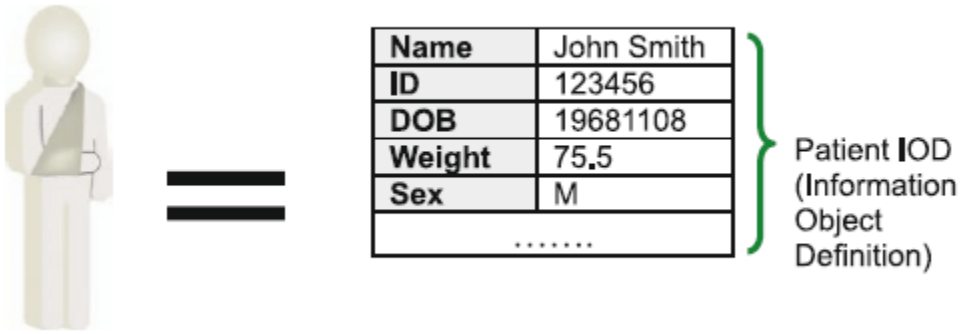
Εικόνα 2-1: Τα κύρια μέρη ενός PACS. Συσκευές ανάκτησης εικόνων (modalities) αποθηκεύουν εικόνες σε ένα ψηφιακό αρχείο. Από εκεί, οι ραδιολόγοι αποκτούν πρόσβαση σε αυτές μέσω απεικονιστικών σταθμών εργασίας

2.1.3.1 Βασικές έννοιες του DICOM

Για να εισάγει την τάξη στο περίπλοκο ιατρικό περιβάλλον, το DICOM χρησιμοποιεί το δικό του λεξιλόγιο, βασισμένο στο μοντέλο του πραγματικού κόσμου (μοντέλο πληροφοριών DICOM). Όλα τα πραγματικά δεδομένα - ασθενείς, μελέτες, ιατρικές συσκευές κ.λπ. - θεωρούνται από το DICOM ως αντικείμενα με αντίστοιχες ιδιότητες ή χαρακτηριστικά[1]. Οι ορισμοί αυτών των αντικειμένων και χαρακτηριστικών είναι τυποποιημένοι σύμφωνα με τα DICOM IODs (Information Object Definitions). Μπορεί κανείς να φανταστεί τα IODs ως συλλογές χαρακτηριστικών που περιγράφουν κάθε συγκεκριμένο αντικείμενο. Το IOD *Ασθενής*, για παράδειγμα, μπορεί να περιγραφεί με το όνομα, τον αριθμό ιατρικού αρχείου (ID), το φύλο, την ηλικία, το βάρος, το εάν καπνίζει και ούτω καθεξής - όσες ιδιότητες χρειάζονται για να μαζευτούν όλες οι κλινικά σχετιζόμενες πληροφορίες του ασθενούς. Υπό μια ευρεία έννοια, ένας ασθενής (όπως και κάθε άλλο

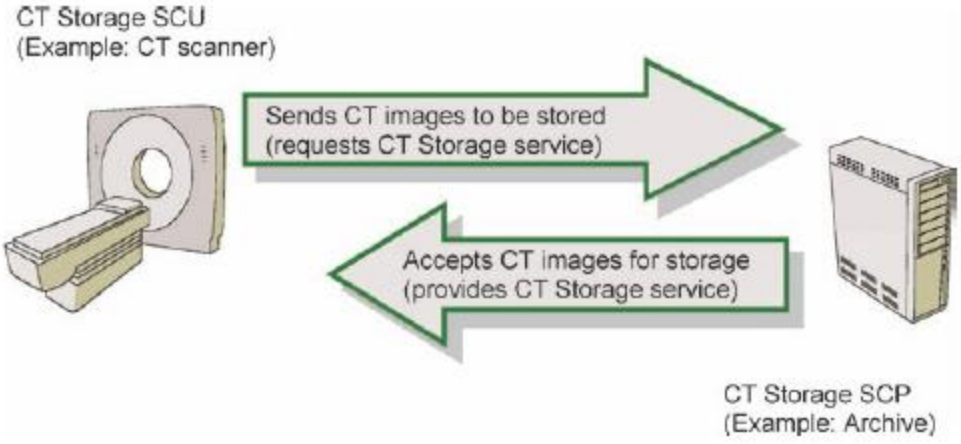
[1] Το συγκεκριμένο μοντέλο προσομοιάζει αυτό των γλωσσών αντικειμενοστραφούς προγραμματισμού

DICOM αντικείμενο) είναι το σύνολο των χαρακτηριστικών από τα οποία αποτελείται, όπως φαίνεται και στην εικόνα 2.2 . Για το σκοπό αυτό, το DICOM διατηρεί μια λίστα με όλα τα τυπικά χαρακτηριστικά (περισσότερα από 2000), για να διασφαλιστεί η συνοχή στην ονομασία χαρακτηριστικών και επεξεργασία. Για παράδειγμα, οι ιδιότητες των ασθενών μας - όνομα, ημερομηνία γέννησης, το φύλο και ούτω καθεξής - περιλαμβάνονται επίσης στο DICOM Data Dictionary. Όλα τα DICOM τα χαρακτηριστικά διαμορφώνονται σύμφωνα με τους τύπους αντιπροσώπευσης τιμών (VR) 27 αντίστοιχων σε ημερομηνίες, ώρες, ονόματα και ούτω καθεξής.



Εικόνα 2-2: Από πραγματικές οντότητες στα DICOM IODs. Κάθε IOD είναι μια συλλογή γνωρισμάτων

Μόλις τα πραγματικά δεδομένα μοντελοποιηθούν ως δεδομένα γνωρισμάτων DICOM, μπορούν να μεταδοθούν και να επεξεργαστούν μεταξύ διαφόρων συσκευών και λογισμικών (Application Entities – AEs, όπως ονομάζονται κατά το πρότυπο). Το DICOM αναπαριστά αυτού του είδους τις διαδικασίες χρησιμοποιώντας ένα μοντέλο προσανατολισμένο σε υπηρεσίες (service-rendering model): οι εφαρμογές DICOM παρέχουν υπηρεσίες η μία στην άλλη. Επειδή κάθε υπηρεσία συνήθως περιέχει κάποιου είδους ανταλλαγή δεδομένων (τυπικά πάνω από δίκτυο υπολογιστών), απορρέει με φυσικό τρόπο ο συσχετισμός συγκεκριμένων ειδών υπηρεσιών με τα δεδομένα (IODs) που αυτά επεξεργάζονται. Το DICOM ονομάζει αυτούς τους συσχετισμούς Service-Object Pairs (SOPs) και τους ομαδοποιεί σε SOP Classes. Για παράδειγμα, αποθηκεύοντας μια αξονική τομογραφία από έναν ψηφιακό αξονικό τομογράφο σε ένα PACS αντιστοιχεί στο λεγόμενο CT Storage SOP, όπως απεικονίζεται στην Εικόνα 2. Στο συγκεκριμένο παράδειγμα, η αξονική τομογραφία (ως εικόνα) αναπαριστά ένα DICOM IOD.



Εικόνα 2-3: DICOM υπηρεσίες

2.1.3.2 Το λεξικό DICOM

Στην ουσία, το λεξικό DICOM (DICOM Data Dictionary) περιέχει όλα τα συνήθη αντικείμενα δεδομένων (γνωρίσματα) που χρησιμοποιούνται στην «ψηφιακή» ιατρική, μορφοποιημένα σύμφωνα με κάποιο από τα 27 διαθέσιμα για το σκοπό αυτό VRs. Για την οργάνωση και ταξινόμηση της πληθώρας αυτής γνωρισμάτων, τα τελευταία διαχωρίζονται πρώτα σε αριθμημένες συλλογές με βάση γενικές ομοιότητες. Οι συλλογές αυτές τώρα αποτελούνται από τα μεμονωμένα στοιχεία/γνωρίσματα. Έτσι, σε κάθε αντικείμενο (γνώρισμα ή στοιχείο) αντιστοιχεί ένα μοναδικό ζεύγος τιμών που έχει τη μορφή (group, element) (γνωστό και ως αναγνωριστικό-tag).

Τόσο τα σύνολα όσο και τα στοιχεία αριθμούνται με δεκαεξαδικούς αριθμούς. Ο Πίνακας 1 περιέχει ένα απόσπασμα του λεξικού. Όπως φαίνεται, η πρώτη στήλη περιλαμβάνει τη δεκαεξαδική τιμή του αναγνωριστικού. Η δεύτερη αντιστοιχεί στην ονομασία του εκάστοτε στοιχείου και ίσως έχει τη μεγαλύτερη σημασία, καθώς επεξηγεί ποια δεδομένα από τον πραγματικό κόσμο πρέπει να αναπαρασταθούν από το συγκεκριμένο στοιχείο. Ξεκάθαρα, τα αναγνωριστικά προσδιορίζουν αμφιμονοσήμαντα τα ονόματα και μπορούν να χρησιμοποιηθούν εναλλάξ για την αναφορά σε κάποιο στοιχείο, ωστόσο τα πρώτα είναι πιο «συμπαγή», σταθερού και μικρού μεγέθους και πιο κοντά στη λογική του υπολογιστή (λόγω και δεκαεξαδικής μορφής). Όλες οι εφαρμογές DICOM (συμπεριλαμβανομένων και των αρχείων ιατρικής εικόνας) αναφέρονται στα στοιχεία αυτά κάνοντας χρήση των εν λόγω αναγνωριστικών. Η στήλη VR καθορίζει τη μορφή του κάθε στοιχείου, όπως παρουσιάστηκε και ανωτέρω. Για παράδειγμα, το στοιχείο «Patient's Birth Date» με tag (0010, 0030), πρέπει να έχει μορφή DA, δηλαδή μια συμβολοσειρά οχτώ χαρακτήρων της μορφής YYYYMMDD (όπου οι χαρακτήρες «YYYY» αντιστοιχούν στο έτος, οι «MM» στο μήνα και, τέλος, οι «DD» στην ημέρα). Η πολλαπλότητα τιμής στοιχείου,

γνωστής και ως VM (Value Multiplicity), καθορίζει το εάν το σχετικό στοιχείο δύναται να περιλαμβάνει μία μόνο τιμή ή πολλαπλές. Για παράδειγμα, το στοιχείο «Other Patient's Names» με tag (0010,1001) μπορεί προφανώς να περιλαμβάνει περισσότερα του ενός ονόματα, για αυτό και η πολλαπλότητα σημειώνεται ως «1 - n» (όπου n οποιοσδήποτε ακέραιος αριθμός). Τέλος, η στήλη «RET» του λεξικού επισημαίνει τα αποσυρθέντα στοιχεία, εκείνα δηλαδή που περιλαμβάνονταν σε προηγούμενες εκδόσεις του προτύπου και θα πάψουν να υποστηρίζονται από μελλοντικές επανεκδόσεις του. Τα στοιχεία αυτά δεν μπορούν να οριστούν ξανά και ο ρόλος τους έχει ανατεθεί σε νέα, καλύτερα σχεδιασμένα στοιχεία.

(Group, Element) tag	Attribute (data element) name	VR	VM	Retired status
(0008,0001)	*Length to End*			*RET*
(0008,0005)	Specific Character Set	CS	1–*n*	
	...			
(0010,0010)	Patient Name	PN	1	
(0010,0020)	Patient ID	LO	1	
(0010,0021)	Issuer of Patient ID	LO	1	
(0010,0030)	Patient's Birth Date	DA	1	
(0010,0032)	Patient's Birth Time	TM	1	
(0010,0040)	Patient's Sex	CS	1	
	...			
(0010,1000)	Other Patient IDs	LO	1–*n*	
(0010,1001)	Other Patient Names	PN	1–*n*	
	...			
(FFFE,E00D)	Item Delimitation Item		1	
(FFFE,E0DD)	Sequence Delimitation Item		1	

Πίνακας 1: Μερικές γραμμές από το λεξικό DICOM

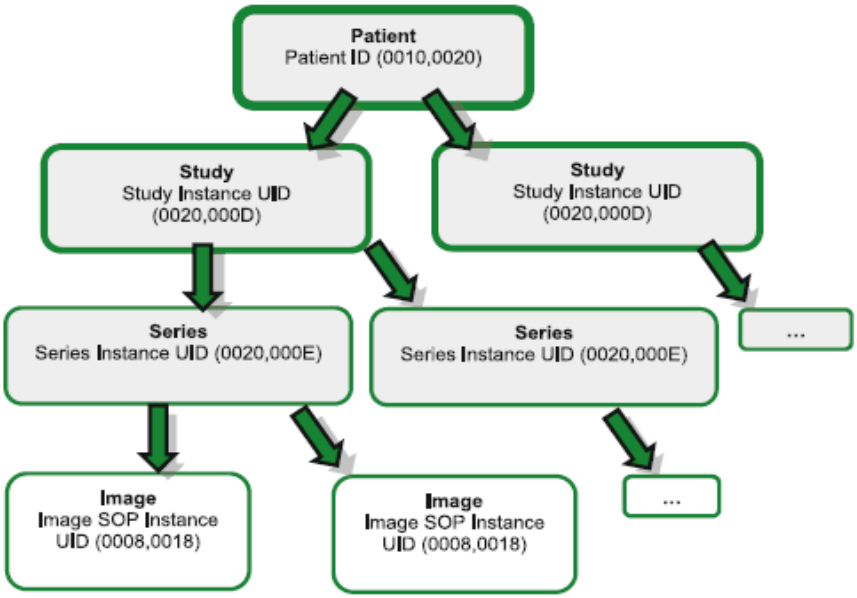
Το λεξικό DICOM αποτελεί στην ουσία το αποτέλεσμα της προσπάθειας του προτύπου να αποδομήσει όλη την περίπλοκη πληροφορία του πραγματικού κόσμου στα μικρότερα δυνατά «ατομικά» κομμάτια (data elements), κωδικοποιημένα με έναν από τους 27 διαφορετικούς τύπους VR. Ένα, λοιπόν, αντικείμενο DICOM (DICOM object) δεν είναι τίποτα άλλο από μια συλλογή τέτοιων στοιχείων – δεν υπάρχει ξεχωριστή DICOM

επικεφαλίδα (“DICOM header”) και DICOM εικόνα (“DICOM image”), όπως πολλοί θέλουν να πιστεύουν. Για παράδειγμα, ας θεωρηθεί μια ψηφιακή ιατρική εικόνα, η οποία περιλαμβάνει χαρακτηριστικά όπως πλάτος, ύψος, χρωματική παλέτα (color palette), ημερομηνία δημιουργίας, τιμές εικονοστοιχείων κτλ. Όλα αυτά τα χαρακτηριστικά μπορούν να βρεθούν μέσα στο λεξικό και θα «μεταφραστούν» σε DICOM data elements, κάθε ένα με το δικό του tag και τιμή. Η ακολουθία αυτών των μεταφρασμένων στοιχείων που περιγράφουν πλήρως την εικόνα στην ολότητά της αποτελεί το αντικείμενο DICOM της εικόνας.

2.1.3.3 Η ιεράρχηση πληροφοριών κατά DICOM

Αναφέρθηκε ήδη η παρουσία DICOM λεξικού δεδομένων το οποίο περιέχει πάνω από 2000 λήμματα/γνωρίσματα και διαδραματίζει κομβικό ρόλο στην αντιστοίχιση δεδομένων πραγματικού κόσμου με το πρότυπο. Εντούτοις, ο αριθμός αυτός είναι σημαντικά μεγάλος για να αφεθεί στην περιγραφή οντοτήτων με τυχαίο και αδόμητο τρόπο. Για το σκοπό αυτό, το DICOM ακολουθεί την ιεραρχία Ασθενής-Μελέτη-Σειρά-Εικόνα (Patient-Study-Series-Image hierarchy, βλ. Εικόνα 2-4):

- Ένας ασθενής μπορεί να έχει πραγματοποιήσει πολλαπλές μελέτες
- Κάθε μελέτη μπορεί να περιέχει πολλαπλές σειρές εικόνων
- Κάθε σειρά έχει μία ή περισσότερες εικόνες



Εικόνα 2-4: Τα 4 επίπεδα της ιεραρχίας πληροφοριών στο DICOM. Κάθε επίπεδο προσδιορίζεται αμφιμονοσήμαντα από ένα στοιχείο-κλειδί

Αυτή η ιεράρχηση αντανακλά με φυσικό τρόπο τι πραγματικά συμβαίνει στον πραγματικό κόσμο όταν ο ασθενής χρειάζεται να πραγματοποιήσει ιατρικές εξετάσεις. Για παράδειγμα, επισκέπτεται ένα νοσοκομείο ή μια ακτινολογική κλινική όπου μπορεί να υποβληθεί σε ένα σύνολο μελετών/εξετάσεων (π.χ. μαγνητική και αξονική τομογραφία, υπερηχογράφημα, κτλ.), στις οποίες αργότερα δύνανται να προστεθούν και άλλες, αν κριθεί σκόπιμο. Κάθε μελέτη μπορεί να έχει πολλαπλές σειρές εικόνων (π.χ. με και χωρίς σκιαγραφικό) κάθε μία εκ των οποίων περιέχει μία (σε περίπτωση π.χ. υπερηχογραφήματος) ή περισσότερες (σε περίπτωση αξονικής ή μαγνητικής) ιατρικές εικόνες. Τώρα, αν χρειαστεί να ανακτηθούν ή να ταξινομηθούν εικόνες του συγκεκριμένου ασθενούς, αρκεί να χρησιμοποιηθούν χαρακτηριστικά γνωρίσματα του ασθενούς αυτού, καθώς επίσης και των μελετών, σειρών και εικόνων που τον αφορούν.

Για να υλοποιήσει αυτήν την ιεραρχία, το DICOM αποδίδει ένα γνώρισμα-κλειδί σε κάθε επίπεδό της. Για το επίπεδο του ασθενούς, αυτό είναι το Patient ID (όλοι οι ασθενείς πρέπει να έχουν μοναδικά αναγνωριστικά – IDs). Παρόμοια, και τα επόμενα επίπεδα διαθέτουν μοναδικά Study Instance UID, Series Instance UID και SOP Instance UID, αντίστοιχα. Όπως είναι λογικό, τα πεδία των γνωρισμάτων αυτών είναι υποχρεωτικό να υπάρχουν σε κάθε έγκυρο αρχείο DICOM.

Όλες οι εντολές DICOM και τα περισσότερα DICOM γνωρίσματα συνδέονται πάντα με αυτή την ιεράρχηση των τεσσάρων επιπέδων. Για το λόγο αυτό, τα τέσσερα γνωρίσματα-κλειδιά διαδραματίζουν έναν καίριο ρόλο: όπως ακριβώς τα ονόματά τους υπονοούν (UID – Unique Identifiers), αναγνωρίζουν με μοναδικό τρόπο τα δεδομένα τους. Αν, λοιπόν, δύο ιατρικές μελέτες έχουν την ίδια τιμή του πεδίου Study Instance UID, τότε αναμένεται να είναι πανομοιότυπες. Αυτό συμπεριλαμβάνει να έχουν επίσης όμοια πεδία Series Instance UID και SOP Instance UID στα αντίστοιχα ιεραρχικά επίπεδα. Παρομοίως, αν δύο ασθενείς έχουν την ίδια τιμή στο Patient ID, αναμένεται να είναι το ίδιο άτομο.

2.1.3.4 Ιατρικές εικόνες στο DICOM

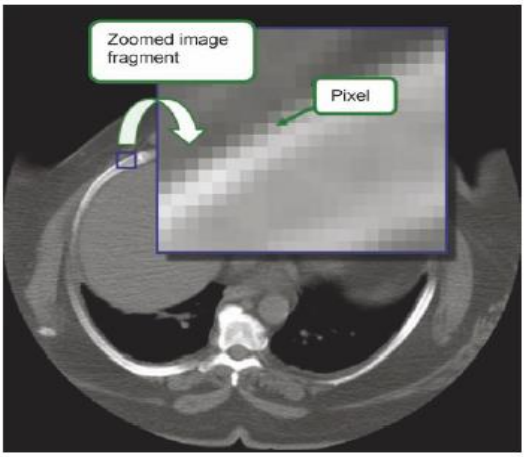
Το πρότυπο DICOM δημιουργήθηκε πρωταρχικά για την ψηφιακή αποθήκευση ιατρικών εικόνων. Οι ψηφιακές εικόνες, γενικά, κατέχουν μερικές γνωστές ιδιότητες, όπως διαστάσεις (πλάτος και ύψος) εκφρασμένες σε εικονοστοιχεία (pixels), βάθος χρώματος σε bits ανά εικονοστοιχείο, όλες εκ των οποίων βρίσκονται στο λεξικό του προτύπου κωδικοποιημένες με κάποιο VR. Το πιο ενδιαφέρον ωστόσο γνώρισμα εικόνας (image attribute) είναι η ίδια η εικόνα – η ακολουθία εικονοστοιχείων δηλαδή που την αποτελούν – αποθηκευμένη στο λεγόμενο «Pixel Data» attribute.

Το DICOM υποστηρίζει ένα μεγάλο εύρος από μορφές εικόνας. Ομαδοποιούνται σε δύο κατηγορίες:

- Ορισμένες κατά DICOM (DICOM-specific), δηλαδή μορφές που χρησιμοποιούνται αποκλειστικά από το πρότυπο. Τυπικά αποτελούν τις πιο παλιές, εισηγμένες από την αρχή της εποχής των υπολογιστών, προτού εξελιχθούν καλύτερες και πιο αποδοτικές μορφές. Ομοιάζουν σε υλοποιήσεις BMPs [3] με διαφόρους τρόπους «συσκευασίας» της πληροφορίας των εικονοστοιχείων.
- Ανεξάρτητες προτυποποιημένες μορφές αποδεκτές από το DICOM. Σε αυτές περιλαμβάνονται διαδεδομένες μορφές όπως JPEG, RLE (Run-Length Encoding), ZIP καθώς και τα λιγότερα γνωστά JPEG2000 και JPEG-LS. Όλα αυτά τα πρότυπα σχετίζονται με διάφορες τεχνικές συμπίεσης, τόσο αναστρέψιμες όσο και μη, γεγονός που τις καθιστά ιδιαίτερα χρήσιμες στην ιατρική απεικόνιση (η μείωση του όγκου των δεδομένων της εικόνας είναι σημαντική). Αυτή η αρθρωτή προσέγγιση, κατά την οποία το κύριο πρότυπο (DICOM) προβλέπει τη χρήση άλλων προτύπων (π.χ. JPEG) για συγκεκριμένες εργασίες (όπως η κωδικοποίηση εικόνας), είναι ιδιαίτερα βολική, συνεπής και έχει πρακτική αξία.

2.1.3.5 DICOM BMPs

Μια ψηφιακή εικόνα, ως γνωστόν, μπορεί να μοντελοποιηθεί ως ένας ορθογώνιος πίνακας εικονοστοιχείων – μικροσκοπικών κουκκίδων διαφορετικού χρώματος – που σχηματίζουν την πραγματική εικόνα. Για παράδειγμα, μία συνήθης εικόνα αξονικής τομογραφίας έχει πλάτος και ύψος 512 εικονοστοιχεία, που σημαίνει ότι περιέχει συνολικά 512 ×512=262,144 εικονοστοιχεία. Αν οι τιμές των εικονοστοιχείων αυτών γραφτούν γραμμή-γραμμή ξεκινώντας από την πάνω αριστερή γωνία της εικόνας σε έναν πίνακα, μπορούν ύστερα να αποθηκευτούν σε ένα αρχείο. Διαισθητικά, αυτό το αρχείο είναι η ακατέργαστη BMP εικόνα (Εικόνα 2-5).



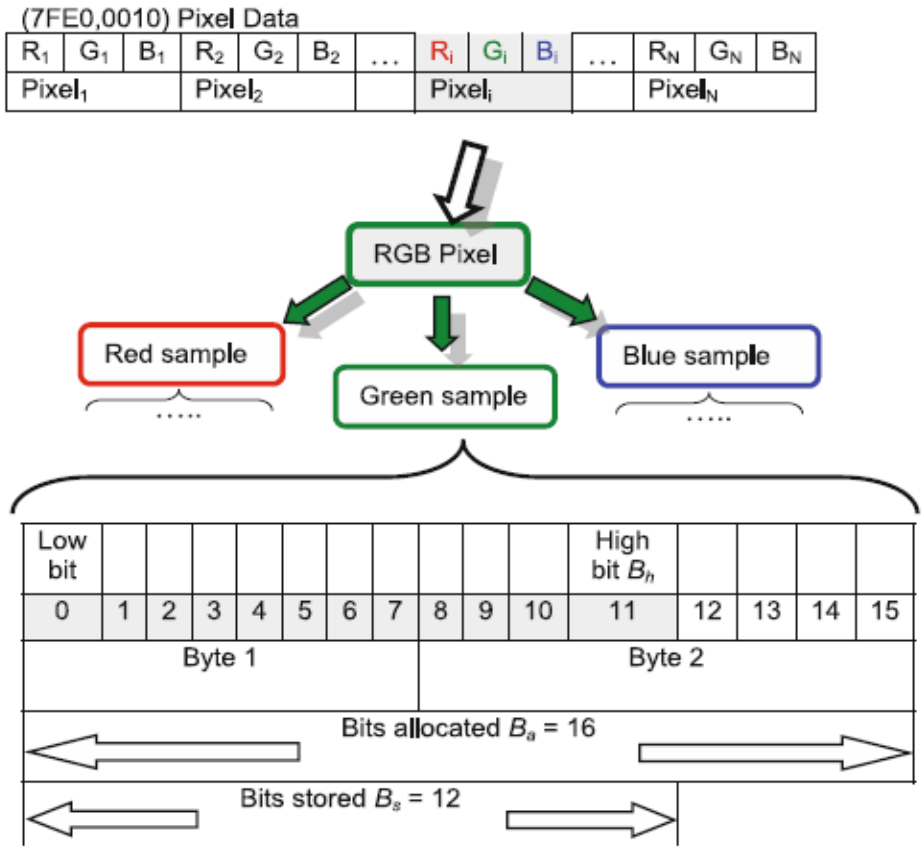
Εικόνα 2-5: Μεγεθύνοντας τα εικονοστοιχεία ιατρικής εικόνας

Απαραίτητα χαρακτηριστικά για την επιτυχή ανάκτηση της πληροφορίας της εικόνας είναι τα κάτωθι:

- Σειρές και στήλες (Rows and Columns): στην ουσία είναι η DICOM ονομασία του πλάτους και ύψους της εικόνας. Το γινόμενό τους (γνωστό και ως χωρική ανάλυση εικόνας – spatial image resolution) ισούται με το συνολικό αριθμό των εικονοστοιχείων που την αποτελούν.
- Δείγματα ανά εικονοστοιχείο (Samples per Pixel): κάθε εικονοστοιχείο μπορεί να αποτελεί μια μίξη διαφόρων τιμών δειγμάτων. Το πιο παραστατικό παράδειγμα είναι ένα έγχρωμο εικονοστοιχείο, το οποίο αποτελείται από τρία ανεξάρτητα χρωματικά δείγματα: κόκκινο, πράσινο και μπλε (γνωστά ως RGB χρωματικός χώρος). Ενώ η «δύναμη» του κάθε δείγματος συνεισφέρει στη φωτεινότητα του εκάστοτε εικονοστοιχείου, η μίξη τους δημιουργεί το χρώμα. Έτσι, αναμειγνύοντας ίση ποσότητα κόκκινου και πράσινου παράγει απόχρωση του κίτρινου τη στιγμή που η ανάμειξη ίσης ποσότητας και των τριών χρωμάτων παράγει απόχρωση του γκρι. Οι ασπρόμαυρες εικόνες, ωστόσο, συνήθως αποθηκεύονται με ένα μονοχρωματικό δείγμα ανά εικονοστοιχείο, αντιστοιχώντας στη φωτεινότητα της απόχρωσης του γκρι που το χαρακτηρίζει. Όταν λοιπόν για μια τέτοια DICOM εικόνα χρησιμοποιούνται δείγματα μεγέθους δύο bytes, παρέχεται η δυνατότητα για απεικόνιση 22×8=65,536 διαφορετικών αποχρώσεων του γκρι. Σε κάθε περίπτωση, η επιλογή του τρόπου δειγματοληψίας παραμένει σταθερή για κάθε εικονοστοιχείο μιας συγκεκριμένης εικόνας – εξαρτάται μόνο από την τεχνική που χρησιμοποιήθηκε κατά την απόκτησή της.
- Αριθμός αποθηκευμένων bits δείγματος εικονοστοιχείου, B_s: “Bits Stored” κατά το DICOM. Σε μια ασπρόμαυρη εικόνα, τα εικονοστοιχεία της οποίας χρησιμοποιούν οκτώ bits, ισχύει B_s=8 και ο αριθμός των διαφορετικών γκρι αποχρώσεών της ισούται με 2^{B_s}=28=256. Αν αυξηθεί ο αριθμός των αποθηκευμένων bits σε B_s=10, τότε θα υποστηρίζει 210=1024 αποχρώσεις του γκρι, κ.ό.κ. Όπως γίνεται αντιληπτό, ο αριθμός των αποθηκευμένων bits είναι αυτός που ορίζει το χρωματικό βάθος της εικόνας (δηλαδή τόσο του αριθμού των χρωμάτων σε μια έγχρωμη εικόνα, όσο και του αριθμού των διαβαθμίσεων του γκρι σε μια ασπρόμαυρη). Υπό μια σκοπιά, το συγκεκριμένο γνώρισμα είναι παρόμοιο με τη χωρική ανάλυση, αλλά στο πεδίο της φωτεινότητας. Είναι δε ξεχωριστής σημασίας για την ιατρικό χώρο όπου απαιτείται ο εντοπισμός ακόμα και απειροελάχιστων διαφοροποιήσεων στο χρώμα και τη φωτεινότητα των εικονοστοιχείων μιας εικόνας.
- Αριθμός δεσμευμένων bits ανά δείγμα εικονοστοιχείου, B_α: ουσιαστικά είναι η τιμή του B_s στρογγυλεμένη προς τα πάνω στην κοντινότερη δύναμη του 2 (έτσι ώστε η τιμή του δείγματος να «χωρά» σε ολόκληρα bytes). Ορίζει την ποσότητα υπολογιστικής μνήμης που απαιτείται για την αποθήκευση ενός χρωματικού δείγματος. Προφανώς, $B_s \leq B_\alpha$.

- Υψηλό bit, B_h: το γνώρισμα που ορίζει το πώς τα αποθηκευμένα bits είναι ευθυγραμμισμένα μέσα στα bits που έχουν δεσμευτεί. Η τιμή του αντιστοιχεί στον αύξων αριθμό του τελευταίου bit που χρησιμοποιείται (το πρώτο bit θεωρείται πως είναι το bit 0).

Η Εικόνα 2-6 συνοψίζει τη δομή ενός έγχρωμου DICOM εικονοστοιχείου. Το τελευταίο αποτελείται από τρία δείγματα (κόκκινο, πράσινο και μπλε) και κάθε δείγμα στο συγκεκριμένο παράδειγμα έχει B_s=12 bits. Επειδή όλα τα δεδομένα αποθηκεύονται από τους υπολογιστές σε bytes (όπου 1 byte = 8 bits), τα 12 bits στρογγυλοποιούνται σε B_a=16, με B_h=11[2]. Η ίδια 16-bits αποθήκευση χρησιμοποιείται και για τα υπόλοιπα δείγματα (κόκκινο και μπλε στη συγκεκριμένη περίπτωση) και ολόκληρη η εικόνα «γράφεται» ως μια ακολουθία από τα δείγματα των εικονοστοιχείων της. Με τον ίδιο ακριβώς σειριακό τρόπο γίνεται η αποθήκευση και ασπρόμαυρων εικόνων, μόνο που αντί για τρία δείγματα ανά εικονοστοιχείο, στην περίπτωση αυτή συναντάται ένα.



Εικόνα 2-6: Αποθήκευση δειγμάτων ανά εικονοστοιχείο έγχρωμης (RGB) εικόνας

[2] Αυτή η λογική όμως επιφέρει σπατάλη αποθηκευτικού χώρου κατά ποσοστό 4⁄16=25% στο συγκεκριμένο παράδειγμα. Για άλλου τύπου εικόνας (π.χ. με B_s=7), το ποσοστό αυτό μπορεί να ανέλθει ακόμα και στο 7⁄16=43,75%! Αυτός είναι ένας από τους λόγους (μαζί και με κάποιους άλλους που θα εξεταστούν στην επόμενη υποενότητα) που εισήχθη η υποστήριξη για ενθυλάκωση συμπιεσμένης εικόνας από το DICOM.

Το DICOM χειρίζεται την πληροφορία της εικόνας όπως οποιαδήποτε άλλη ενθυλακωμένη πληροφορία και όλα τα σημαντικά γνωρίσματά της κωδικοποιούνται σε κάποιας μορφής VR. Στη γλώσσα του DICOM, η ύπαρξή τους είναι υποχρεωτική. Ο Πίνακας 2 παρέχει μια συγκεντρωτική (αλλά όχι εξαντλητική) απογραφή μερικών σημαντικών γνωρισμάτων που περιλαμβάνονται στο λεξικό του DICOM (συμπεριλαμβανομένων των ήδη παρουσιασμένων). Κάποιος με υπόβαθρο στην ψηφιακή απεικόνιση δεν μπορεί παρά να εκτιμήσει την πληρότητα του προτύπου. Για παράδειγμα, όπως υποδηλώνει το στοιχείο με κωδικό (0028,0008), κάποιος μπορεί να αποθηκεύσει μια ακολουθία από καρέ (frames) – στην ουσία βίντεο ψηφιακής μορφής – μέσα σε ένα μόνο αρχείο εικόνας. Επίσης, μέσω του πεδίου (0028,0030) προσδιορίζονται οι φυσικές διαστάσεις των εικονοστοιχείων, κάτι που επιτρέπει την πραγματοποίηση μετρήσεων σε φυσικά μεγέθη (μήκος σε mm, επιφάνεια σε mm^2) πάνω στην ιατρική εικόνα. Γνωρίζοντας τώρα τις διαστάσεις αυτές σε συνδυασμό με την απόσταση μεταξύ των εικόνων που αποτελούν μια σειρά μέσω του πεδίου (0018,0088), είναι δυνατή η επίτευξη πολύπλοκων τρισδιάστατων ανακατασκευών, αφού διατηρούνται τα ακριβή μεγέθη των αντικειμένων.

Tag	Name	VR	VM
(0028,0002)	Samples per Pixel	US	1
(0028,0004)	Photometric Interpretation	CS	1
(0028,0008)	Number of Frames	IS	1
(0028,0010)	Rows	US	1
(0028,0011)	Columns	US	1
(0028,0030)	Pixel Spacing	DS	2
(0028,0100)	Bits Allocated B_a	US	1
(0028,0101)	Bits Stored B_s	US	1
(0028,0102)	High Bit B_h	US	1
(0028,0103)	Pixel Representation	US	1
(7FE0,0010)	Pixel Data	OW/OB	1

Πίνακας 2: Μερικά από τα πιο συνήθη γνωρίσματα εικόνας που περιέχονται στο λεξικό του DICOM

2.1.3.6 Συμπιεσμένες DICOM εικόνες

Θεωρώντας πάλι το προηγούμενο παράδειγμα της εικόνας αξονικής τομογραφίας, έχει αξία να υπολογιστεί η ποσότητα μνήμης που απαιτείται για την αποθήκευσή της.

Καταρχήν, αποτελείται από 262,144 εικονοστοιχεία (όπως έχει υπολογιστεί ήδη). Πρόκειται για ασπρόμαυρη εικόνα, οπότε κάθε εικονοστοιχείο έχει ένα χρωματικό δείγμα. Αν τώρα υποτεθεί (και στην πράξη όντως ισχύει) πως το δείγμα αυτό περιγράφεται με οποιοδήποτε πλήθος bits μεταξύ 8 και 16, τότε συμπεραίνεται πως τα απαιτούμενα bytes ανά δείγμα και, κατά συνέπεια, εικονοστοιχείο είναι 2. Οπότε, για την αποθήκευση ολόκληρης της εικόνας απαιτούνται συνολικά 512×512×2=524,288 bytes ≈0,5 MB. Όμως, όπως δείχθηκε, οι εικόνες αξονικής τομογραφίας παράγονται στα πλαίσια κάποιας μελέτης και κάποιας σειράς, κάθε μία εκ των οποίων μπορεί να περιέχει μερικές δεκάδες έως και εκατοντάδες ιατρικές εικόνες. Αν δε συνυπολογίσει κανείς τη συνεχή βελτίωση των ακτινολογικών και ραδιολογικών μονάδων με αποτέλεσμα την αύξηση της ανάλυσης και του βάθους των παραγόμενων εικόνων, η αποθήκευση και η αποστολή τους – ειδικά παλαιότερα που το υλισμικό ήταν πιο ακριβό και οι ταχύτητες των δικτύων περιοριστικές, αποτελούσε πρόβλημα για οποιοδήποτε οργανισμό επιχειρούσε να επενδύσει σε ένα λειτουργικό και κλιμακώσιμο τηλεακτινολογικό (teleradiological) εξοπλισμό (βλ. Πίνακα 3).

Για να αντιμετωπίσει τους σκοπέλους αυτούς, το DICOM υιοθέτησε την υποστήριξη για συμπίεση εικόνας σχεδόν από τις απαρχές του. Η συμπίεση αυτή εφαρμόζεται στοχευμένα στα δεδομένα εικονοστοιχείων του πεδίου “Pixel Data” και με έξυπνο τρόπο τα αναδιατάσσει σε μια πολύ πιο σύντομη μορφή. Κατά συνέπεια, επιτυγχάνεται και μια παράλληλη σημαντική μείωση του μεγέθους ολόκληρου του αρχείου DICOM – αν αναλογιστεί κανείς πως το μεγαλύτερο κομμάτι του καταλαμβάνεται από τα δεδομένα της εικόνας, που με τη σειρά του εξοικονομεί αποθηκευτικό χώρο και ελαττώνει το χρόνο διαμοιρασμού του πάνω από το δίκτυο. Το ίδιο το πρότυπο, όπως προαναφέρθηκε, δεν υλοποιεί κάποιο δικό του μηχανισμό συμπίεσης, αλλά χρησιμοποιεί κάποιες δοκιμασμένες και δημοφιλείς λύσεις, όπως RLE, JPEG, JPEG2000, JPEG-LS και ZIP. Όλοι οι αλγόριθμοι στους οποίους είναι βασισμένες, έχουν αναπτυχθεί ξεχωριστά και έχουν καταλήξει να αποτελούν ξεχωριστά ISO (International Organization for Standardization) πρότυπα. Το DICOM επέλεξε απλώς να τα υιοθετήσει.

Image modality	Typical image matrix (height, width, bytes per pixel)	Image size, kilobytes (KB)	Typical number of images in a study[a]	Typical study size, megabytes (MB)
NM	128×128×1	16	100	1.5
MR	256×256×2	128	200	25
CT	512×512×2	512	500	250
Ultrasound	600×800×3	1400	500	680
CR	2140×1760×2	7356	4	30
Color 3D reconstructions[b]	1024×1024×3	3000	20	60
Digital mammography	Up to 6400×4800×2	60,000	4	240

Πίνακας 3: Τυπικά μεγέθη ιατρικών εικόνων και μελετών

Η εφαρμογή της συμπίεσης εικόνας μπορεί να διαδραματίσει καταλυτικό ρόλο στην εμφάνισή της αλλά και στη συνολική επίδοση των PACS. Κάθε αλγόριθμος συμπίεσης αναζητά να βρει και να απομακρύνει πλεονάζουσα και επαναληπτική πληροφορία στην προσπάθειά του να μειώσει το μέγεθος των δεδομένων. Η αποδοτικότητά του προσεγγίζεται με το λεγόμενο λόγο συμπίεσης:

$$R_{comp} = \frac{αρχικό\ μέγεθος\ δεδομένων}{συμπισμένο\ μέγεθος\ δεδομένων}$$

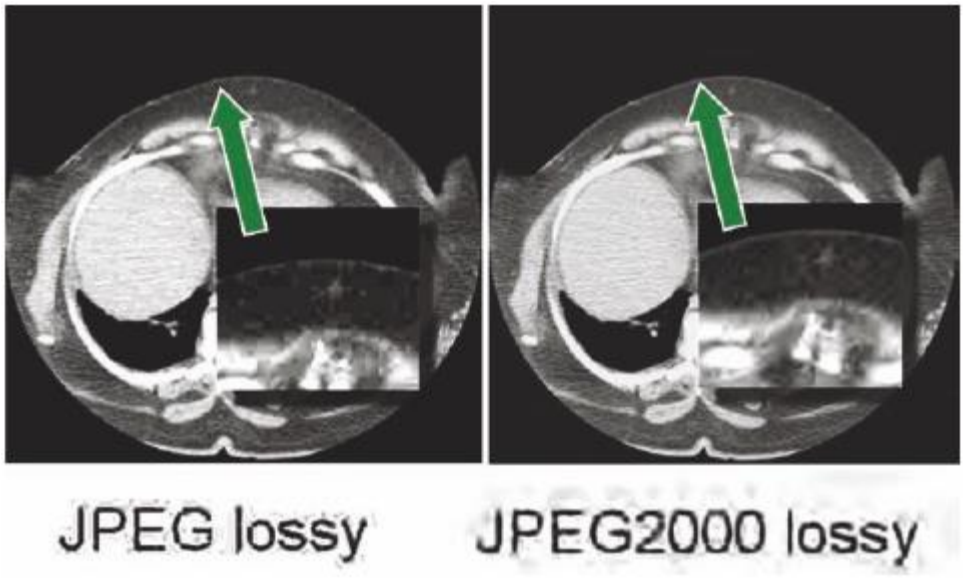
Όσο μεγαλύτερος ο λόγος, τόσο πιο αποτελεσματική συμπίεση επιτυγχάνεται. Κάθε αλγόριθμος προσφέρει τη δική του στρατηγική μεγιστοποίησης του R_{comp}, αλλά εννοιολογικά και πρακτικά, όλες οι διαφορετικές τεχνικές που προσφέρουν ομαδοποιούνται σε μη-απωλεστικές (lossless) και σε απωλεστικές (lossy).

Η μη-απωλεστική συμπίεση δεν τροποποιεί την εικόνα. Όσες φορές και αν συμπιεστεί και στη συνέχεια αποσυμπιεστεί, πάντοτε τα δεδομένα της θα είναι ίδια – δεν παρατηρείται απώλεια πληροφορίας. Αυτό επιτυγχάνεται με έξυπνη ανασύνταξη και μετονομασία των bytes των εικονοστοιχείων. Σε γενικές γραμμές, ένας τυπικός αλγόριθμος μη-απωλεστικής συμπίεσης θα προσπαθήσει να εντοπίσει την επαναληπτικότητα των συχνότερων τιμών και να τις αντικαταστήσει με συντομότερα σύμβολα (π.χ. αν πολλαπλά εικονοστοιχεία μιας ασπρόμαυρης εικόνας έχουν την τιμή 1000, αυτή μπορεί να αντικατασταθεί με το σύμβολο “α”). Ωστόσο, οι μετασχηματισμοί αυτοί μπορούν να χρησιμοποιηθούν μόνο μέχρι ενός σημείου και για μια μέση ιατρική εικόνα, ο λόγος R_{comp}

βρίσκεται κάπου μεταξύ δύο και τέσσερα – νούμερο αποδεκτό αλλά όχι εντυπωσιακό αν συγκριθεί με το αντίστοιχο μιας απωλεστικής προσέγγισης.

Η δεύτερη εναλλακτική πετυχαίνει σημαντικά μεγαλύτερο βαθμό συμπίεσης, αλλά με το κόστος – όπως φανερώνει και το όνομα – της απώλειας αρχικής πληροφορίας. Η θυσία αυτή γίνεται έτσι ώστε να εισαχθεί τεχνηέντως πλεονασμός δεδομένων τον οποίο και θα εκμεταλλευτεί περαιτέρω ο αλγόριθμος μη-απωλεστικής συμπίεσης. Με λίγα λόγια, αν η μη-απωλεστική συμπίεση αξιοποιεί τα εικονοστοιχεία με ίσες τιμές, η απωλεστική επεκτείνει αυτή την αξιοποίηση και σε εικονοστοιχεία σχεδόν ίσων τιμών με ένα δύσκολα αντιληπτό περιθώριο λάθους. Με τον τρόπο αυτό, οι σημερινοί απωλεστικοί αλγόριθμοι μπορεί να επιτύχουν λόγο συμπίεσης ίσο ακόμα και με 100, αν και σε μια μέση περίπτωση περιορίζονται μεταξύ του 10 και του 20. Η τιμή του λόγου στην απωλεστική συμπίεση εξαρτάται από την τιμή που ορίζεται ως επιτρεπτό σφάλμα. Προφανώς, και το σφάλμα αυτό δεν μπορεί να αυξάνεται επ' αόριστον προκειμένου να πετυχαίνεται συνεχώς καλύτερος βαθμός συμπίεσης. Από ένα σημείο και μετά, στη συμπιεσμένη εικόνα αρχίζουν και εισάγονται ορατές αλλοιώσεις (artifacts), όπως απεικονίζεται και στην Εικόνα 2-7. Η ισορροπία μεταξύ υψηλού R_{comp} και ορατής παραμόρφωσης της εικόνας έχει γίνει πλέον μια «μορφή τέχνης» από μόνη της. Σημαντικοί παράγοντες για τη διαμόρφωσή της είναι οι εξής:

- Η απωλεστική συμπίεση μπορεί να οδηγήσει σε νομικές διαφορές. Αν πρόκειται να χρησιμοποιηθούν εικόνες με τέτοιου είδους συμπίεση, το DICOM και ο FDA (Food and Drug Administration) απαιτούν την υποχρεωτική επισήμανσή τους.
- Ζητήματα διάγνωσης με τη βοήθεια υπολογιστή (Computer-Aided Diagnosis – CAD). Καθώς οι τεχνικές CAD γίνονται δημοφιλέστερες, οι υπολογιστές και το λογισμικό τους εμπλέκονται όλο και περισσότερο στην ανάλυση ιατρικής εικόνας. Σφάλματα αόρατα στο ανθρώπινο μάτι μπορεί να είναι ορατά για το λογισμικό, υπονομεύοντας την αποτελεσματικότητά του.



Εικόνα 2-7: Αποτελέσματα υπερβολικής απωλεστικής συμπίεσης σε εικόνες και γραμματοσειρές. Στα αριστερά, η συμπίεση JPEG παράγει τα λεγόμενα blocking artifacts. Στα δεξιά, η συμπίεση JPEG2000 παράγει έντονη θολότητα.

Στην Εικόνα 2-7 αντιπαρατίθενται τα δύο πιο δημοφιλή πρότυπα συμπίεσης στο χώρο της ιατρικής απεικόνισης, το JPEG και το JPEG2000. Το πρώτο χρησιμοποιεί αλγορίθμους βασισμένους στο μετασχηματισμό DCT (Discrete Cosine Transform) ενώ το δεύτερο και μεταγενέστερο αλγορίθμους βασισμένους σε wavelets. Ο DCT εκφράζει μια πεπερασμένη ακολουθία σημείων δεδομένων (data points) μέσω ενός αθροίσματος συνημιτονοειδών συναρτήσεων διαφορετικών συχνοτήτων. Στην ουσία, είναι παρόμοιος με το DFT (Discrete Fourier Transform), αλλά χρησιμοποιεί μόνο πραγματικούς αριθμούς. Το wavelet είναι μια μαθηματική συνάρτηση που χρησιμοποιείται στη διαίρεση μιας δοθείσας άλλης συνάρτησης ή σήματος σε διαφορετικά κλιμακούμενα συστατικά (scale components). Συνήθως κάποιος μπορεί να αναθέσει ένα εύρος συχνοτήτων σε κάθε κλιμακούμενο συστατικό, το οποίο κατόπιν μπορεί να μελετηθεί με μια ανάλυση που ταιριάζει στην κλίμακά του. Ένας wavelet μετασχηματισμός είναι η αναπαράσταση μιας συνάρτησης από wavelets.

Το JPEG κάνει χρήση του 8 x 8 DCT, ενώ το JPEG2000 του DWT (Discrete Wavelet Transformation). Ο τελευταίος παρέχει όχι μόνο καλύτερη ενεργειακή συμπύκνωση (οπότε και υψηλότερο κέρδος συμπίεσης), αλλά και κλιμακωσιμότητα της ανάλυσης – αφού οι συντελεστές του wavelet μπορούν να διαχωριστούν σε διαφορετικές αναλύσεις, είναι εφικτό να εξαχθεί μιας χαμηλότερης ανάλυσης εικόνα χρησιμοποιώντας μόνο τους αναγκαίους συντελεστές. Το JPEG ακόμα, κατατέμνει την εικόνα σε blocks των 8 x 8 ή 16 x 16 στο πεδίο του χώρου και στη συνέχεια εφαρμόζει τους υπόλοιπους μετασχηματισμούς. Εφόσον ακολουθεί αυτή τη λογική της ανεξάρτητης κωδικοποίησης των blocks, από ένα

σημείο και μετά γίνονται ορατά τα blocking artifacts της παραπάνω εικόνας. Σε αντίθεση, το JPEG2000 πραγματοποιεί την κατάτμηση στο πεδίο των wavelet και σε συνδυασμό με τον αντίστοιχο μετασχηματισμό, δεν εμφανίζει παρόμοια προβλήματα. Επίσης, ενώ με το JPEG είναι δυνατή η προβολή μιας εικόνας σε συγκεκριμένη μόνο ανάλυση ορισμένη κατά τη κωδικοποίηση, το JPEG2000 επιτρέπει τη μερική αποσυμπίεση των wavelets που περιέχει, προκειμένου ο χρήστης να προβάλει, αν επιθυμεί την εικόνα σε χαμηλότερη ανάλυση. Τέλος, ένα ακόμα σημαντικό χαρακτηριστικό που προσδίδει η χρήση των wavelets στο JPEG2000 είναι οι ROI (Region of Interest) δυνατότητες, δηλαδή να μπορεί να προβάλλει κάποιος μια συγκεκριμένη περιοχή της εικόνας σε υψηλότερη ποιότητα από ότι την υπόλοιπη, με συνέπεια τη μείωση της απαιτούμενης μνήμης αλλά και του χρόνου πρόσβασης. Από όλα τα παραπάνω, γίνεται εύκολα κατανοητό το πλεονέκτημα που προσδίδει η χρήση JPEG2000 κωδικοποίησης για τις ιατρικές εικόνες, τόσο κατά την αποθήκευσή τους, όσο και κατά τη μεταφορά τους πάνω από το δίκτυο.

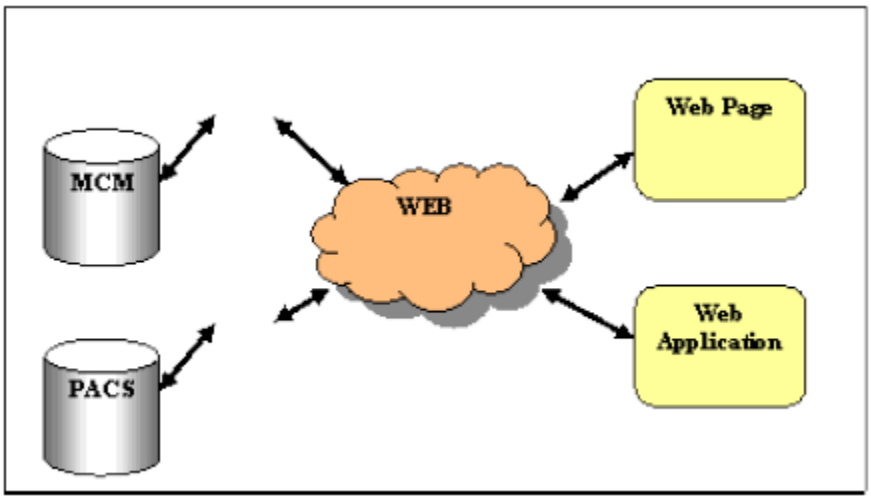
2.1.3.7 Web Access to DICOM Objects (WADO)

Σήμερα, τα περισσότερα DICOM αντικείμενα αρχειοθετούνται σε ειδικές αποθήκες (repositories), όπως PACS και MCM[3], για συγκεκριμένες χρονικές περιόδους. Όταν ένα ιατρικό πληροφοριακό σύστημα επιθυμεί να συνδεθεί σε μια αποθήκη και να ανακτήσει DICOM πληροφορία, πρέπει να είναι «εξοικειωμένο» με το συγκεκριμένο, συνήθως κλειστό (proprietary) πρωτόκολλο που η αποθήκη χρησιμοποιεί. Το γεγονός αυτό συνεπάγεται την επένδυση σημαντικών προσπαθειών κατά την εξέλιξη εφαρμογών με δυνατότητα απομακρυσμένης σύνδεσης σε τέτοιου είδους αποθήκες δεδομένων. Επιπρόσθετα, οι εν λόγω εφαρμογές γενικά υπολείπονται σε ευελιξία, καθότι δεν μπορούν εύκολα να επεκταθούν για την υποστήριξη περισσοτέρων αποθηκών.

Για να αντιμετωπιστούν αυτοί οι σκόπελοι και να καταστεί δυνατή η πρόσβαση σε DICOM πληροφορία από οπουδήποτε στον ιστό, δημιουργήθηκε το πρότυπο WADO (για την ακρίβεια αποτελεί μέρος του ιδίου του DICOM, καθώς περιγράφεται στην παράγραφο PS 3.18-2004 των προδιαγραφών του). Το τελευταίο ορίζει μια υπηρεσία ιστού (web-based service) για την ανάκτηση και παρουσίαση αποθηκευμένων αντικειμένων DICOM, όπως εικόνες και αναφορές ιατρικής απεικόνισης (medical imaging reports) και έχει ως σκοπό τη διανομή αποτελεσμάτων και εικόνων σε επαγγελματίες υγείας. Στην ουσία, παρέχει έναν απλό μηχανισμό ώστε η πρόσβαση αυτή να καθίσταται δυνατή από σελίδες HTML ή αρχεία XML διαμέσου HTTP/HTTPS, κάνοντας παράλληλη χρήση των μοναδικών αναγνωριστικών που παρέχει το DICOM. Τα δεδομένα μπορούν να ανακτηθούν είτε σε κάποια μορφή έτοιμη προς παρουσίαση (π.χ. σε μορφή εικόνας JPEG ή PNG) είτε στη φυσική τους. Για

[3] Το MCM (Medical Content Manager) είναι ένα σύστημα της IBM για αποθήκευση ιατρικού περιεχομένου που απαιτεί μακροχρόνια διαχείριση και αρχειοθέτηση [60].

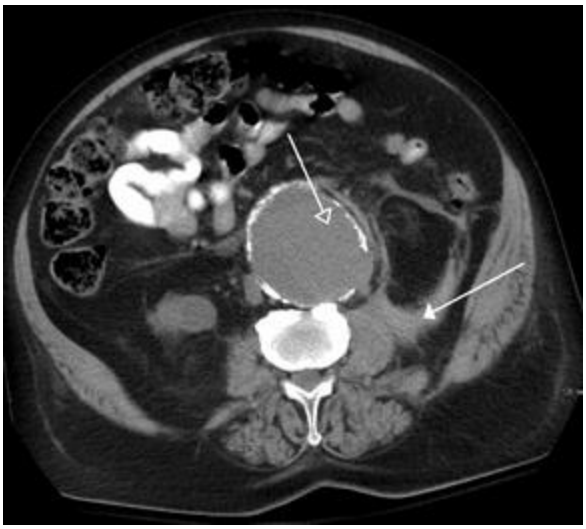
παράδειγμα, ένας ιατρός-χρήστης δύναται να εκτελέσει απομακρυσμένα αναζήτηση σε μια αποθήκη που συμμορφώνεται με το WADO για ένα συγκεκριμένο ασθενή (μέσω του πεδίου Patient ID) και κατόπιν να πραγματοποιήσει λήψη μιας ολόκληρης μελέτης (μέσω του Series Instance UID) στην οποία έχει υποβληθεί στο παρελθόν.



Εικόνα 2-8: Το WADO ως υπηρεσία

2.2 Ιατρική απεικόνιση

Η ιατρική απεικόνιση είναι η τεχνική και η διαδικασία δημιουργίας οπτικής αναπαράστασης του εσωτερικού του ανθρωπίνου σώματος για κλινική ανάλυση και ιατρική παρέμβαση. Η ιατρική απεικόνιση αναζητά να αποκαλύψει εσωτερικές δομές κάτω από το δέρμα και τα κόκκαλα, με σκοπό να συμβάλλει στη διάγνωση και τη θεραπεία παθήσεων. Ακόμα, εγκαθιδρύει μια βάση δεδομένων με στοιχεία ανθρώπινης ανατομίας και φυσιολογίας επιτρέποντας έτσι την πιο εύκολη αναγνώριση ανωμαλιών. Παρότι η απεικόνιση οργάνων και ιστών που έχουν αφαιρεθεί μπορεί να γίνει για ιατρικούς λόγους, τέτοιες πρακτικές θεωρούνται μέρος του κλάδου της παθολογίας.



Εικόνα 2-9: Αξονική τομογραφία αποκαλύπτει ρήξη ανευρύσματος της κοιλιακής αορτής

Ως επιστημονικός κλάδος και εξεταζόμενος κάτω από ένα ευρύτερο πρίσμα, η ιατρική απεικόνιση αποτελεί μέρος της βιολογικής απεικόνισης και ενσωματώνει την ακτινολογία (radiology), η οποία χρησιμοποιεί τεχνολογίες απεικόνισης όπως ακτινογραφία (X-ray radiography), μαγνητική τομογραφία (magnetic resonance imaging), υπέρηχος (medical ultrasonography ή ultrasound), ελαστογραφία (elastography), θερμογραφία (thermography), τομογραφία εκπομπής ποζιτρονίων (positron emission tomography) κ.ά.

2.3 Εφαρμογές απεικόνισης ιατρικών εικόνων

Η Βιοϊατρική Τεχνολογία γνωρίζει ιδιαίτερη άνθιση λόγω των αυξημένων απαιτήσεων για ένα ποιοτικότερο και πιο σωστά δομημένο σύστημα υγείας. Μέχρι και το 2010, έχουν πραγματοποιηθεί πέντε δισεκατομμύρια μελέτες ιατρικής απεικόνισης [4]. Εξαιτίας λοιπόν αυτής της διάδοσης της ιατρικής εικόνας τα τελευταία χρόνια αλλά και της προτυποποίησης αυτής (βλ. Ενότητα 2.1.3), έχει εμφανιστεί μια ευρεία γκάμα desktop και web εφαρμογών που επιτρέπουν την προβολή και επεξεργασία της. Για τους σκοπούς αυτής της διπλωματικής εργασίας αναφέρονται ενδεικτικά μερικές από τις πιο γνωστές:

2.3.1 OsiriX

Η εφαρμογή OsiriX είναι ανοικτό λογισμικό επεξεργασίας εικόνων παραγόμενων από εξοπλισμό ιατρικής απεικόνισης (MRI, CT, PET, PET-CT, SPECT-CT, Ultrasound, κτλ.) για την πλατφόρμα Macintosh. Εξελίσσεται και συντηρείται από την Pixmeo, μια εταιρία με βάση τη Γενεύη. Είναι πλήρως συμβατό με το πρότυπο αρχείων DICOM (επέκτασης .dcm) αλλά και το πρωτόκολλο επικοινωνίας του, οπότε μπορεί να λειτουργήσει παράλληλα και ως σταθμός εργασίας DICOM PACS. Έχει σχεδιαστεί συγκεκριμένα για πλοήγηση και απεικόνιση πολυδιάστατων εικόνων πολλαπλών modalities (2D, 3D, 4D και 5D) χρησιμοποιώντας όλες τις σύγχρονες πρακτικές, όπως πολυεπίπεδη ανασύνθεση (multiplanar reconstruction), απεικόνιση επιφάνειας (surface rendering), απεικόνιση τρισδιάστατων μοντέλων (volume rendering), κτλ. Τέλος, να σημειωθεί ότι είναι γραμμένο σε Objective-C, η αρχιτεκτονική του υποστηρίζει την επέκταση των δυνατοτήτων του σύμφωνα με τις ανάγκες του χρήστη μέσω plug-ins, ενώ είναι από τους ελάχιστους viewers που είναι εγκεκριμένος από τον αμερικανικό οργανισμό FDA (Food and Drug Administration).

2.3.2 RadiAnt

Το RadiAnt είναι μια standalone εφαρμογή για την πλατφόρμα των Windows η οποία διατίθεται δωρεάν (freeware) αλλά, σε αντίθεση με το OsiriX, ο κώδικάς της δεν είναι ανοικτός. Είναι και αυτή συμβατή με το πρότυπο DICOM κάτι που σημαίνει πως μπορεί να ανοίξει και να προβάλλει οποιοδήποτε αρχείο .dcm, εντούτοις δεν υποστηρίζει αλληλεπίδραση με κάποιο PACS. Οι δυνατότητες που παρέχει στο χρήστη είναι βασικές και δε φτάνουν σε κανένα βαθμό το εύρος του OsiriX (έλλειψη προηγμένων τεχνικών rendering, φίλτρων, image fusion), αλλά η ταχύτητά του είναι αξιοσημείωτη μιας και εκμεταλλεύεται πλήρως τις σύγχρονες πολυπύρηνες αρχιτεκτονικές, η διεπαφή της υποδειγματική και ο χρόνος εξοικείωσης μαζί της μικρός.

2.3.3 LEADTOOLS HTML5 Zero-footprint Medical Viewer

Το προϊόν αυτό είναι ένα web application αποτελούμενο από μια ισχυρή συλλογή από βιβλιοθήκες σε JavaScript και RESTful Web Services που παρέχουν τη δυνατότητα για προβολή, διαχείριση και επεξεργασία DICOM εικόνας από PACS. Αυτό σημαίνει πως δεν υποστηρίζει σε αντίθεση με τις προηγούμενες εφαρμογές, την πρόσβαση σε αρχεία DICOM από εξωτερικές συσκευές αποθήκευσης, αλλά το γεγονός ότι είναι γραμμένο για να εκτελείται σε περιβάλλον browser, το καθιστά προσβάσιμο ανεξαρτήτως πλατφόρμας

(platform-independent) χωρίς να θυσιάζει τίποτα από άποψη λειτουργικότητας και δυνατοτήτων. Ακόμα σημειώνεται πως διατίθεται έναντι πληρωμής.

2.4 Ερευνητικό υπόβαθρο

Σε ερευνητικό επίπεδο, η τρισδιάστατη επεξεργασία και αναπαράσταση ιατρικών δεδομένων έχει απασχολήσει σημαντικά την ιατρική κοινότητα. Η πλειοψηφία των εργασιών που εκτελούνται στη ραδιολογία αναπαρίστανται ως δισδιάστατη πληροφορία, από συμβατικές εικόνες X-ray μέχρι τις πιο προηγμένες υπολογιστικών και μαγνητικών τομογράφων. Στα πρώτα στάδια της ραδιολογίας, όταν πρόκυπτε ανάγκη για τρισδιάστατη απεικόνιση, αυτή πραγματοποιούνταν με στερεοτακτικές συσκευές που απεικόνιζαν δύο εικόνες, μία για κάθε μάτι, με ελάχιστη αλλαγή (περίπου 5 μοίρες) μεταξύ των δύο εικόνων. Όταν όμως ξεκίνησε η αξονική απεικόνιση με υπολογιστικούς τομογράφους στις ΗΠΑ, οι πληροφορίες απεικόνισης έγιναν διαθέσιμες σε ψηφιακή μορφή και στη συνέχεια ανακαλύφθηκαν τρόποι απόκτησης ογκομετρικών πληροφοριών από τις δισδιάστατες εικόνες.

Επόμενο βήμα ήταν οι εισαγωγή μεθόδων για τη μετεπεξεργασία των δεδομένων υπολογιστικού τομογράφου: Προβολή μέγιστης έντασης (MIP – Maximum Intensity Projection), Πολυεπίπεδη ανασύνθεση (MPR – Multi Planar Reconstruction) και Απεικόνιση σκιασμένης επιφάνειας (3DSSD – Surface Shaded Display). Το MPR αποτελεί την πιο διαδεδομένη μέθοδο μετεπεξεργασίας και στις πιο πολλές περιπτώσεις τη μοναδική επιλογή, η οποία χρησιμοποιήθηκε και στο πλαίσιο αυτής της εργασίας.

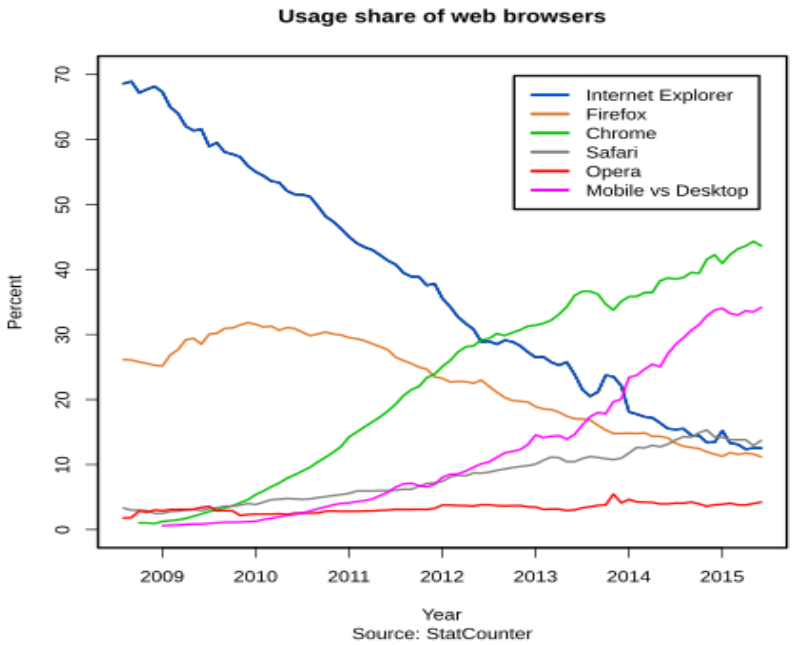
Αντίστοιχα, για την τρισδιάστατη ανακατασκευή εμφανίστηκαν οι μέθοδοι 3D Απεικόνιση σκιασμένης επιφάνειας (3DSSD – Surface Shaded Display) και Ανακατασκευή απεικόνισης όγκου (VR - Volume Rendering reconstruction). Η πρώτη μέθοδος αναγνωρίζει τους ιστούς από την πυκνότητα τους ή χειροκίνητα σχεδιάζοντας το περίγραμμα του οργάνου, δείχνοντας επί της ουσίας την επιφάνεια των οργάνων ως ένα αδιαφανές αντικείμενο. Εντούτοις, χρησιμοποιείται πολύ σπάνια διότι απαιτεί αρκετά εκτεταμένη δουλεία από το χρήστη [5]. Η πλέον ενδεδειγμένη λοιπόν λύση είναι το VR, το οποίο κάνει τους υπολογισμούς του λαμβάνοντας υπόψιν τη συνεισφορά κάθε voxel στο δείγμα (βλ. Ενότητα 4.2). Αυτή είναι η μέθοδος η οποία αξιοποιήθηκε στο πλαίσιο της εργασίας αυτής.

3 Τεχνολογικό υπόβαθρο

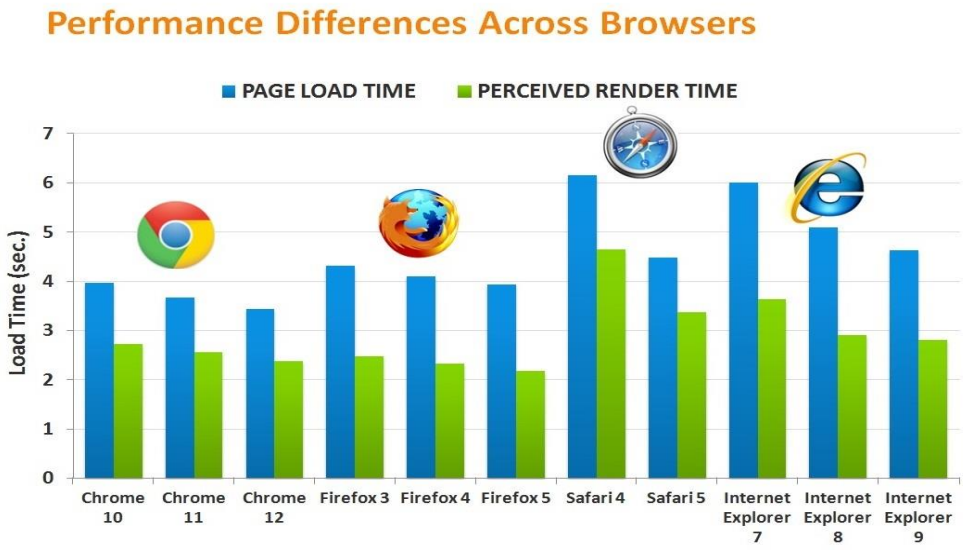
3.1 Τεχνολογίες Πληροφορικής και Επικοινωνιών

3.1.1 Φυλλομετρητές Ιστού (Web Browsers)

Ο web browser ή φυλλομετρητής ιστοσελίδων είναι μια μορφή λογισμικού που επιτρέπει στον χρήστη του να προβάλλει και να αλληλοεπιδρά με κείμενα, εικόνες, βίντεο, μουσική, και άλλες πληροφορίες, συνήθως αναρτημένες σε μια ιστοσελίδα ενός ισοτόπου στον Παγκόσμιο Ιστό ή σε ένα τοπικό δίκτυο. Οι φυλλομετρητές χρησιμοποιούν τη γλώσσα μορφοποίησης HTML για την προβολή των ιστοσελίδων, πράγμα που οδηγεί σε διαφορετική εμφάνιση της ίδιας σελίδας σε διαφορετικό browser. Οι πλοηγοί Web ουσιαστικά αποτελούν λογισμικό πελάτη του δικτυακού πρωτοκόλλου επιπέδου εφαρμογών HTTP [6]. Όπως φαίνεται από το διάγραμμα των Εικόνων 3-1 και 3-2, τα τελευταία χρόνια ο πλέον δημοφιλής web browser είναι αυτός που εισήγαγε η Google το 2008, ο οποίος δεν είναι άλλος από τον Chrome. Λαμβάνοντας υπόψιν τη δημοτικότητα του Chrome καθώς και την προσωπική προτίμηση ως προς αυτόν, επιλέχτηκε ως περιβάλλον εξέλιξης της εφαρμογής – συστήματος. Ένας δεύτερος λόγος επιλογής του Google Chrome ως περιβάλλοντος εξέλιξης της εφαρμογής είναι η απόδοση και η σταθερότητα που παρουσιάζει σε σχέση με τους άλλους browsers, το οποίο είναι ιδιαίτερα σημαντικό για εφαρμογές Τηλεϊατρικής.



Εικόνα 3-1: Ποσοστά χρήσης των web browsers παγκοσμίως



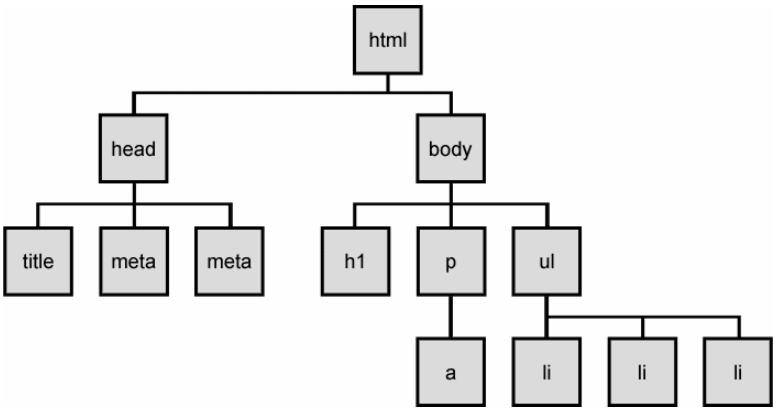
Εικόνα 3-2: Απόδοση Διάφορων Web Browsers

3.1.2 Web Applications ως Ολοκληρωμένες Εφαρμογές

Μια Εφαρμογή Ιστού (Web Application) έχει ως σκοπό την αλληλεπίδραση του χρήστη όχι μόνο με απλή προβολή κειμένου και εικόνας αλλά με πιο πλούσια και πιο περίπλοκη λειτουργικότητα. Η γλώσσα HTML γράφεται με τη μορφή στοιχείων (HTML elements) αποτελούμενων από tags, περικλειόμενα από τους χαρακτήρες ‘<’, ‘>’, όπως φαίνεται και στην Εικόνα 3-3. Ο κώδικας μετατρέπεται από τον browser σε μια δενδρική δομή αποτελούμενη από JavaScript αντικείμενα-κόμβους, ή όπως αλλιώς ονομάζεται DOM (Document Object Model, βλ. Εικόνα 3-4). Ο σκοπός της μετατροπής αυτής είναι η παροχή μιας προγραμματιστικής διεπαφής για την εκτέλεση σεναρίων (scripting), όπως αφαίρεση, πρόσθεση, αντικατάσταση και τροποποίηση του «ενεργού» HTML εγγράφου χρησιμοποιώντας JavaScript. Στα HTML στοιχεία μπορούν να αποδοθούν γνωρίσματα με τη μορφή ζευγών ονόματος-τιμής (key-value pairs), ώστε να καταστεί δυνατή η παραμετροποίησή τους. Στοιχεία που αντιστοιχούν σε κόμβους στο DOM μπορούν επίσης να δημιουργηθούν δυναμικά χρησιμοποιώντας μεθόδους που παρέχει η JavaScript, όπως η createElement(). Το νόημα όλων αυτών είναι πως JavaScript και HTML είναι στενά συνδεδεμένες, υπό την έννοια ότι ο προγραμματισμός σε HTML μπορεί να επιτευχθεί με χρήση JavaScript.

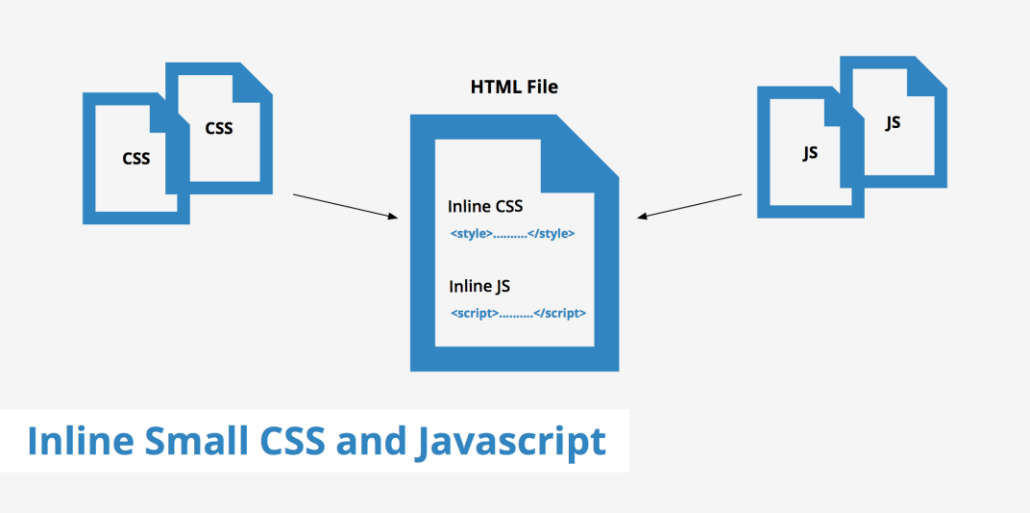
```
<html>
    <head>
        <title>A Basic Webpage</title>
    </head>
    <body>
        <h1>My Heading</h1>
        <p>My paragraph.</p>
    </body>
</html>
```

Εικόνα 3-3: Γραφική μορφή ενός απλού HTML κειμένου



Εικόνα 3-4: Document Object Model μιας web σελίδας

Η δημιουργία ιστοσελίδων μόνο με HTML ωστόσο κρίνεται ανεπαρκής για τα σημερινά δεδομένα του δικτυακού προγραμματισμού. Οι σύγχρονοι ιστότοποι μπορούν να εννοηθούν ως ένας συνδυασμός δομής (structure), ύφους (style) και διαδραστικότητας (interactivity) [7]. Υπεύθυνες για τα τρία αυτά διαφορετικά στοιχεία είναι οι τεχνολογίες HTML, CSS και JavaScript, αντίστοιχα, οι οποίες από κοινού συνεισφέρουν στη διανομή πλούσιου περιεχομένου σελίδων ιστού. Η HTML μέσω κατάλληλων tag μπορεί να συμπεριλάβει πλέον CSS και JavaScript αρχεία (tag <style> για CSS και tag <script> για JavaScript, βλ. Εικόνα 3-5).



Εικόνα 3-5: Οι τεχνολογίες που σε συνδυασμό δημιουργούν εφαρμογές δικτύου

Η γλώσσα CSS (Cascading Style Sheet) δημιουργήθηκε και προτάθηκε από την κοινοπραξία W3C (World Wide Web Consortium), τον κύριο διεθνή οργανισμό προτύπων για το Web, τον Αύγουστο του 1996 [8]. Γενικά χρησιμοποιείται για να παραμετροποιήσει την εμφάνιση του HTML περιεχομένου, προσθέτοντας γνωρίσματα όπως στυλ γραμματοσειράς, χρώμα, διαστάσεις κτλ. Παλιότερα, κάτι τέτοιο ήταν εφικτό και με χρήση μόνο HTML, αλλά ήταν μια επίπονη προγραμματιστική διαδικασία, καθώς για κάθε στοιχείο έπρεπε να γραφτούν όλα τα γνωρίσματα «με το χέρι». Με την εισαγωγή της CSS, σε ένα ή περισσότερα στοιχεία μπορεί να ανατεθεί μια κλάση (class attribute) και ο κώδικας που είναι υπεύθυνος για τους κανόνες εμφάνισης της κλάσης να συνταχτεί μία φορά και η αντιστοίχιση με τα στοιχεία να γίνεται αυτόματα από το browser.

Το πρόβλημα όμως της διαδραστικότητας των ιστοσελίδων παρέμεινε. Η λύση δόθηκε το Δεκέμβριο του 1995. Η JavaScript δημιουργήθηκε από την Netscape Communications Corporation [9]. Με την εισαγωγή της JavaScript οι ιστοσελίδες είναι πιο δυναμικές καθώς υπάρχει η δυνατότητα για την αλληλεπίδραση με το D.O.M. και για προσθήκη λειτουργικότητας: φόρμες κειμένου, κουμπιά, εκτέλεση υπολογισμών και αλγορίθμων, προσθήκη event handlers (ειδικές συναρτήσεις που καλούνται όταν συμβεί κάποιο γεγονός), επικοινωνία με τον εξυπηρετητή και γενικότερα όλες τις δυνατότητες που προσφέρει μια πλήρης (κατά Turing) γλώσσα προγραμματισμού. Ένα επιπλέον πλεονέκτημα αυτής της προσέγγισης είναι ότι ο κώδικας JavaScript μεταφράζεται και εκτελείται από τον browser του χρήστη μέσω των JavaScript engines που υπάρχουν στον πυρήνα των browsers. Έτσι η εκτέλεση δεν βασίζεται στην ταχύτητα του δικτύου άλλα αποκλειστικά στο τοπικό λογισμικό και υλικό.

Η πλέον διαδεδομένη αρχιτεκτονική οργάνωσης μιας web page είναι η HTML σελίδα να περιέχει τα κατάλληλες αναφορές σε εξωτερικά αρχεία (<script > και <style>

αντίστοιχα) και ο αντίστοιχος πραγματικός κώδικας να βρίσκεται σε διαφορετικά αρχεία, πράγμα που εξυπηρετεί την αυτονομία των επιμέρους στοιχείων, την αποσφαλμάτωση, τη συντήρηση και την εύκολη αναγνωσιμότητα. Η τάση γι' αυτή την οργάνωση είναι πλέον αποκλειστική, γεγονός που έχει οδηγήσει στην ανάπτυξη ειδικών εργαλείων τα οποία βοηθούν το έργο του προγραμματιστή. Φυσικά κάποια από αυτά χρησιμοποιήθηκαν για την παρούσα πλατφόρμα.

3.1.3 Web Browser Συμβατότητα

Ως web browser συμβατότητα ορίζεται η δυνατότητα μιας ιστοσελίδας, διαδικτυακής εφαρμογής, ενός σεναρίου, ή μιας μορφοποίησης HTML να λειτουργεί με τον ίδιο τρόπο σε όλους τους browsers που υπάρχουν στο εμπόριο.

Το κέρδος αυτής της προσέγγισης είναι ότι η ιστοσελίδα είναι διαθέσιμη σε όλο το κοινό ανεξάρτητα από τον φυλλομετρητή, ενώ δεν υπάρχει κάποια μείωση απόδοσης μεταξύ διαφορετικών browsers [13]. Παρ' όλες τις προσπάθειες που έχουν γίνει για μια ομοιογένεια και προτυποποίηση των browsers ακόμα υπάρχουν διαφοροποιήσεις μεταξύ τους οι οποίες μπορούν να οδηγήσουν ακόμα και σε ανεξήγητες συμπεριφορές. Γι' αυτό δίνεται η δυνατότητα στον προγραμματιστή να ρωτήσει τον Web Browser αν υποστηρίζει κάποια χαρακτηριστικά και να αποφευχθούν έτσι λάθη. Οι τεχνολογίες που χρησιμοποιήθηκαν στην παρούσα εφαρμογή (WebGL, WebRTC, HTTP requests) είναι πλέον εγκατεστημένες στους περισσότερους browsers και έτσι δεν αντιμετωπίστηκαν περιπτώσεις μη συμβατότητας των τεχνολογιών.

3.1.4 JavaScript

Για ολόκληρη την εφαρμογή χρησιμοποιήθηκε η JavaScript (ο server υλοποιήθηκε με Node.js που είναι γραμμένο σε JavaScript, βλ. Ενότητα 3.1.10), οπότε κρίνεται απαραίτητο να γίνει μια πιο αναλυτική αναφορά σε αυτήν. Όπως είδαμε στα παραπάνω κεφάλαια τα προγράμματα clients έχουν μεγάλο φόρτο εργασίας, πράγμα που σημαίνει ότι ο κώδικας JavaScript είναι εκτενέστερος και υπολογιστικά απαιτητικότερος. Αυτό σημαίνει ότι οι μηχανές JavaScript πρέπει να είναι γρήγορες και αποτελεσματικές.

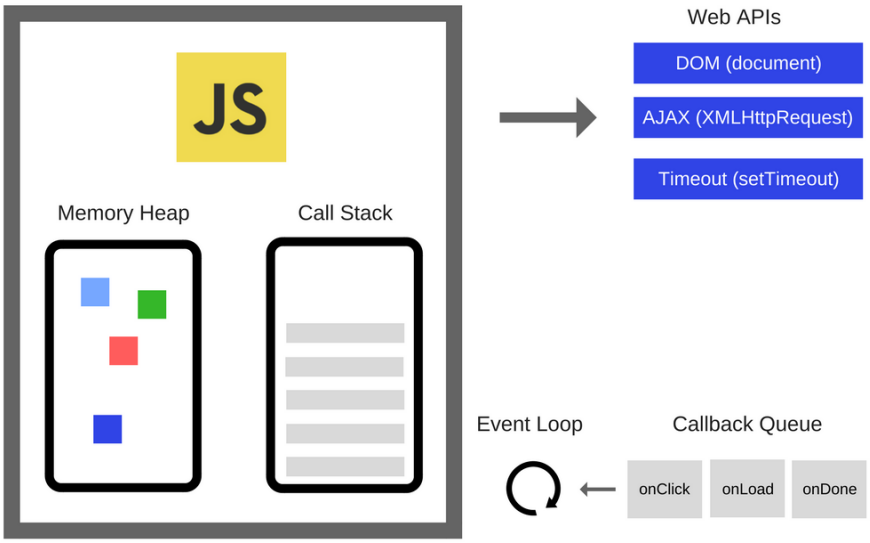
3.1.4.1 Δυναμικό Σύστημα Τύπων και Ταυτοχρονισμός

Η γλώσσα JavaScript είναι μονονηματική (single-threaded), έτσι πολλαπλά scripts δεν είναι σε θέση να τρέξουν παράλληλα. Ένα δεύτερο χαρακτηριστικό της JavaScript, που δυνητικά μπορεί να καθυστερήσει την εκτέλεση ενός προγράμματος, είναι το δυναμικό

σύστημα τύπων και ο δυναμικός έλεγχος. Αυτό σημαίνει ότι ο συμπερασμός τύπου των μεταβλητών (int, float, string) γίνεται κατά τη διάρκεια της εκτέλεσης, οπότε και πιθανά σφάλματα ανιχνεύονται σε αυτό το στάδιο και όχι νωρίτερα. Τα δύο παραπάνω χαρακτηριστικά της γλώσσας γίνονται άμεσα αντιληπτά στην περίπτωση που σε μία ιστοσελίδα-εφαρμογή θα χρειαστεί να τρέξουν πολλά διαφορετικά scripts, κάθε ένα από τα οποία θα έχει να επεξεργαστεί έναν μεγάλο όγκο δεδομένων. Αυτό θα έκανε την εφαρμογή αργή ή ακόμα και μα αποκρίσιμη. Όμως οι σύγχρονοι browsers είναι σε θέση να αποκρύπτουν την πολυπλοκότητα αυτή από τον χρήστη και να παραμένουν αποκρίσιμοι.

3.1.4.2 Ο Βρόχος Συμβάντων (event loop)

Στη JavaScript, σχεδόν όλες οι λειτουργίες εισόδου/εξόδου (I/O) εκτελούνται α-σύγχρονα, χωρίς μπλοκάρισμα (non-blocking execution). Σε αυτές περιλαμβάνονται HTTP αιτήσεις, ενέργειες πάνω σε βάσεις δεδομένων, εγγραφές και αναγνώσεις από το σκληρό δίσκο. Το μοναδικό νήμα ζητά από το περιβάλλον εκτέλεσης να πραγματοποιήσει μια ενέργεια, παρέχοντάς του μια συνάρτηση επανάκλησης (callback function), συνεχίζοντας με την εκτέλεση άλλων εργασιών. Όταν η ενέργεια που ζητήθηκε προηγουμένως ολοκληρωθεί, ένα μήνυμα εισάγεται σε μια ουρά μαζί με ένα παρεχόμενο callback. Κάποια στιγμή στο μέλλον, το μήνυμα αυτό θα αφαιρεθεί από την ουρά και η συνάρτηση επανάκλησης θα εκτελεστεί. Αυτό το διαδραστικό, πλήρως ασύγχρονο, μοντέλο μπορεί να είναι γνώριμο στους προγραμματιστές που αναπτύσσουν λογισμικό διεπαφής χρήστη – εκεί που συμβάντα όπως το πάτημα ενός κουμπιού ή η κύλιση του παραθύρου μπορούν να προκύψουν οποιαδήποτε στιγμή και πρέπει να «εξυπηρετηθούν», αλλά διαφέρει σημαντικά από το σύγχρονο μοντέλο αίτησης-απόκρισης που συναντάται σε τυπικές υλοποιήσεις εφαρμογών εξυπηρετητή. Αυτή η απεμπλοκή του καλούντος από την απάντηση που αυτός αναμένει, επιτρέπει στο περιβάλλον εκτέλεσης της JavaScript να ασχοληθεί με άλλες διεργασίες ενώ «περιμένει» την ασύγχρονη ενέργεια να διεκπεραιωθεί και να έρθει η στιγμή να εκκινήσει το callback. Αυτή η ουρά – αόρατη στον προγραμματιστή – στην οποία τα μηνύματα αποθηκεύονται προσωρινά μαζί με τα αντίστοιχα εγγεγραμμένα callbacks, ονομάζεται βρόχος συμβάντων (βλ. Εικόνα 3-6).



Εικόνα 3-6: Βρόχος συμβάντων της JavaScript

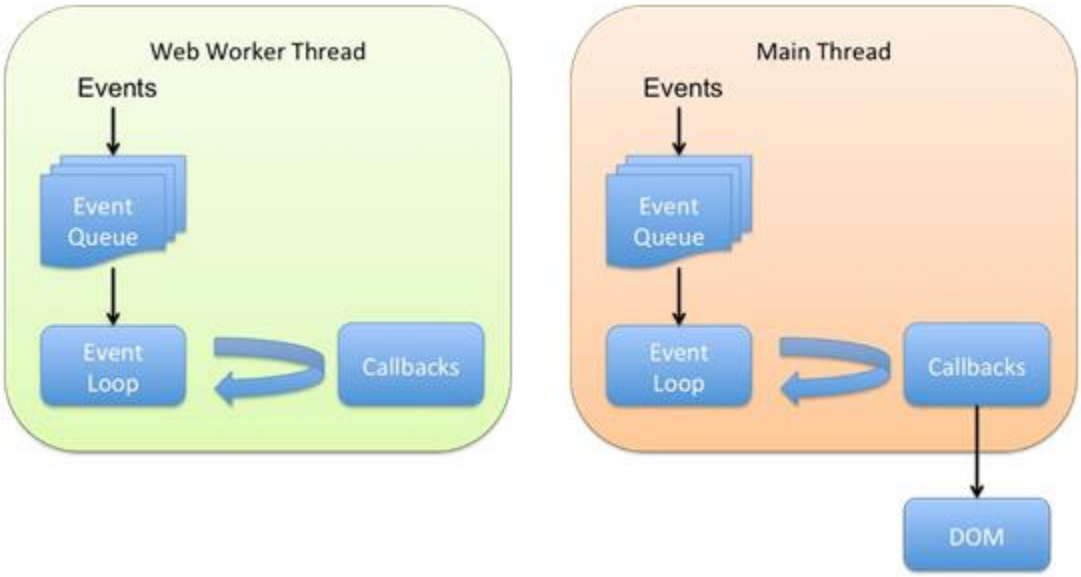
3.1.4.3 Run-to-completion logic

Η πρακτική που ακολουθεί το περιβάλλον της JavaScript είναι η πλήρης επεξεργασία κάθε μηνύματος προτού συνεχίσει με το επόμενο. Αυτό προσφέρει κάποιες ελκυστικές ιδιότητες κατά το σχεδιασμό της λογικής των προγραμμάτων, συμπεριλαμβανομένης της εγγύησης της εγκυρότητας των δεδομένων κατά τη διάρκεια της εκτέλεσης συναρτήσεων. Παρατηρείται δηλαδή διαφορά με το μοντέλο της C, κατά το οποίο, όταν μία συνάρτηση ή κομμάτι κώδικα εκτελείται μέσα σε κάποιο νήμα, το σύστημα έχει τη δυνατότητα να διακόψει την εκτέλεσή τους και να μεταφέρει τον έλεγχο σε κάποιο άλλο νήμα.

Βέβαια, υπάρχουν και μειονεκτήματα στην προσέγγιση αυτή. Το σπουδαιότερο εκ των οποίων έχει να κάνει με το εάν ένα μήνυμα χρειαστεί σημαντικό χρονικό διάστημα για την επεξεργασία του, όλη η εφαρμογή καθίσταται ανίκανη να διαχειριστεί οποιαδήποτε αλληλεπίδραση με το χρήστη. Δεν είναι σπάνια, ακόμα και σήμερα, η εμφάνιση ειδοποίησης με τη μορφή ξεχωριστού παραθύρου από το browser, σύμφωνα με την οποία «η εκτέλεση κάποιου script καθιστά την εφαρμογή μη αποκρίσιμη». Οι προγραμματιστές για να υπερκεράσουν τους περιορισμούς αυτούς προσπαθούσαν να μειώσουν όσο γίνεται το χρόνο επεξεργασίας που απαιτούσαν ή, αν αυτό δεν ήταν δυνατό, να «μοιράσουν» το φόρτο σε μικρότερα μηνύματα, τα οποία τοποθετούσαν εκ νέου στο βρόχο συμβάντων χρησιμοποιώντας ειδικά API calls, όπως set-Timeout() [10] και setInterval() [11].

3.1.4.4 Web Workers

Μεγάλη πρόοδο επιτεύχθηκε στο συγκεκριμένο τομέα με την έλευση των Web Workers. Σύμφωνα με τις επίσημες προδιαγραφές, οι browsers πλέον καλούνται να υλοποιήσουν ένα API για την παρασκηνιακή εκτέλεση scripts από τις εφαρμογές ιστού. Με τον τρόπο αυτό, υπολογιστικά απαιτητικές εργασίες μπορούν να μεταφερθούν σε δικό τους ανεξάρτητο νήμα εκτέλεσης με το δικό του ανεξάρτητο βρόχο συμβάντων, χωρίς να υπάρχει το ενδεχόμενο να «μπλοκάρουν» τη διεπαφή της εκάστοτε εφαρμογής με το χρήστη, η οποία «τρέχει» στο κύριο νήμα (Εικόνα 3-7).



Εικόνα 3-7: Web worker και κύριο νήμα εκτέλεσης

Η επικοινωνία μεταξύ του κυρίου νήματος (main or UI thread) και των Web workers λαμβάνει χώρα μέσω ενός μοντέλου συμβάντων (event model) και της μεθόδου postMessage(). Ανάλογα με τον τύπο και την έκδοση του browser καθώς και τον τρόπο κλήσης της μεθόδου, η τελευταία μπορεί να λειτουργήσει με δύο τρόπους. Οι περισσότεροι browsers υλοποιούν το λεγόμενο structured cloning αλγόριθμο, ο οποίος επιτρέπει το πέρασμα πολύπλοκων τύπων δεδομένων από και προς τους workers, όπως File, Blob, ArrayBuffer και JSON αντικείμενα. Ωστόσο, τα αντικείμενα αυτά αντιγράφονται πριν το πέρασμά τους (κάτι αντίστοιχο με το πέρασμα κατά τιμή στη C), γεγονός που εισάγει καθυστέρηση στην όλη διαδικασία. Για το λόγο αυτό, αναπτύχθηκε μια δεύτερη τεχνική που αξιοποιεί τα λεγόμενα transferrable objects. Χρησιμοποιώντας την, δεδομένα μεταφέρονται από το περιβάλλον του κυρίου νήματος σε αυτό του παρασκηνιακού χωρίς να αντιγράφεται τίποτα με σημαντικό αντίκτυπο στην επίδοση. Ομοιάζει με το πέρασμα

κατ’ αναφορά της C, με μια μικρή αλλά σημαντική διαφορά: όταν ένα αντικείμενο επιχειρηθεί να «περαστεί» από το κύριο νήμα σε κάποιο worker, το κύριο χάνει την αναφορά που κρατούσε στο αντικείμενο αυτό, και το ανάποδο. Αυτό συμβαίνει λόγω της σχεδιαστικής επιλογής των δημιουργών του προτύπου να διαχωρίσουν πλήρως το χώρο μνήμης των διαφόρων νημάτων μιας εφαρμογής έτσι ώστε να εξασφαλίσουν την αποφυγή φαινομένων, όπως data races και deadlocks.

Να σημειωθεί ακόμα πως από το περιβάλλον ενός Web Worker είναι αδύνατη η χρησιμοποίηση όλων των χαρακτηριστικών και APIs της JavaScript. Ο κύριος περιορισμός έχει να κάνει με την αδυναμία αλληλεπίδρασης με το DOM, μιας και το JavaScript αντικείμενο που είναι υπεύθυνο για το ρόλο αυτό (document object) δε μοιράζεται με τα νήματα που τρέχουν στο παρασκήνιο. Για τον ίδιο λόγο δεν υπάρχει πρόσβαση και σε τεχνολογίες που εξετάστηκαν παραπάνω (WebGL και Canvas). Παρ’ όλα αυτά, οι workers είναι σε θέση να επικοινωνήσουν με το δίκτυο, μέσω του XMLHttpRequest API [12].

Αν αναλογιστεί κανείς την πρόοδο που έχει γίνει σήμερα στο κομμάτι του υλισμικού και ειδικά των πολυπύρηνων αρχιτεκτονικών πάνω στις οποίες είναι βασισμένη η πλειονότητα των συσκευών στις οποίες έχουν πρόσβαση οι χρήστες, ένα πρότυπο σαν και αυτό αποτελούσε το επόμενο εύλογο βήμα. Το αντίκτυπο λοιπόν που είχε η εισαγωγή των Web Workers στην ανάπτυξη εφαρμογών ιστού, αντικατοπτρίζεται στην πληθώρα σεναρίων χρήσης που ήδη καλύπτουν:

- Δημιουργία, γραμματική ανάλυση και επεξεργασία αρχείων PDF.
- Συμπίεση/αποσυμπίεση εικόνων σε προσαρμοσμένη μορφή αρχείου (JPEG, BMP, PNG).
- Υπολογισμός A* μονοπατιού σε περιβάλλον διαδραστικών παιχνιδιών.
- Αναζήτηση λέξης σε βάση δεδομένων για ορθογραφικό έλεγχο σε επεξεργαστή κειμένου.
- Φόρτωση και γραμματική ανάλυση πόρων μέσα σε παιχνίδι.
- Πραγματοποίηση πολύπλοκων υπολογισμών στο παρασκήνιο και απομακρυσμένη καταγραφή των αποτελεσμάτων.

Η κοινότητα όμως των κατασκευαστών browsers δε μένει εκεί αλλά προσπαθεί συνεχώς να εκμεταλλευτεί την αυξημένη δυνατότητα παραλληλίας που τα μηχανήματα σήμερα προσφέρουν με σκοπό πάντα την παροχή συνεχώς βελτιωμένης εμπειρίας χρήσης. Προς τη κατεύθυνση αυτή, λοιπόν, έχουν προτυποποιηθεί τελευταία και δύο καινούρια παρόμοια APIs: Shared Workers [13] και Service Workers [14]. Το πρώτο από αυτά αφορά την πρόσβαση σε νήματα παρασκηνίου από διαφορετικά παράθυρα του browser, <iframe> tags ή και άλλα ακόμα νήματα, ενώ το δεύτερο εισάγει νέου τύπου workers με σκοπό την παροχή πλούσιας εμπειρίας χρήσης ακόμα και σε κατάσταση εκτός σύνδεσης, δυνατότητας περιοδικού συγχρονισμού με το backend στο παρασκήνιο, την υποστήριξη push

notifications από τον εξυπηρετητή, και άλλων προηγμένων χαρακτηριστικών που συνεχώς εμπλουτίζονται.

3.1.5 Single Page Applications

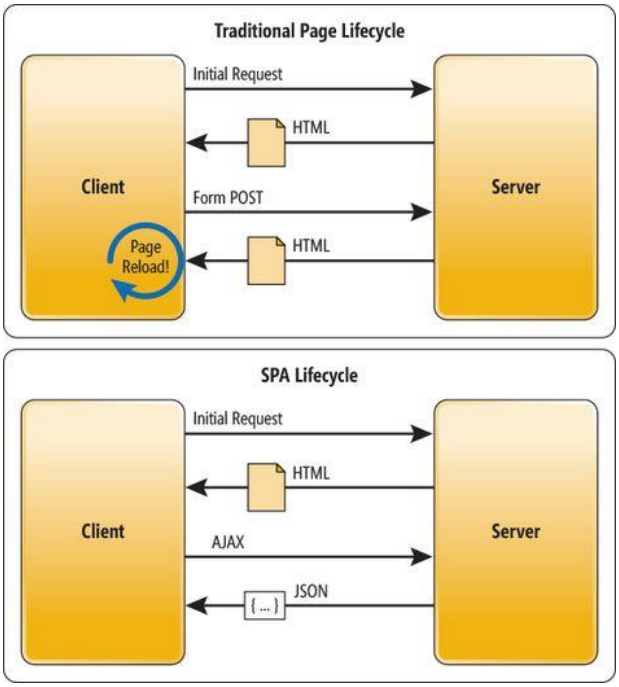
3.1.5.1 Τα Προγράμματα Πελάτες Γίνονται πιο Σύνθετα

Όπως έχει ήδη αναφερθεί από την αρχική εισαγωγή των browsers ως περιβάλλον εκτέλεσης του δικτυακού προγραμματισμού, η πρόοδος είναι αλματώδης. Ενώ η αρχική προσέγγιση ήταν οι HTML σελίδες να χρησιμοποιούνται ως απλό μέσο για την παρουσίαση στατικών διαδικτυακών σελίδων, με την εμφάνιση του CGI (Common Gateway Interface) ξεκίνησε η περίοδος των δυναμικών ιστοσελίδων, δηλαδή σελίδων που δημιουργούνταν ανάλογα με το συγκεκριμένο χρήστη σε κάποιο ευρύτερο πλαίσιο, με την βοήθεια κάποιου backend συστήματος, δηλαδή servers υλοποιημένων σε PHP/ASP/JSP, κλπ όπου οι πελάτες (thin clients) απλά απεικόνιζαν το περιεχόμενο αυτό.

3.1.5.2 AJAX (Asynchronous JavaScript and XML)

Η μεγάλη αλλαγή όμως στο διαδικτυακό προγραμματισμό επιτεύχθηκε με την είσοδο των AJAX (Asynchronous JavaScript and XML) τεχνικών οι οποίες πλέον μετατόπισαν τον έλεγχο της ροής και των πληροφοριών από τους εξυπηρετητές στους πελάτες. Ο client side προγραμματισμός είχε αναβαθμιστεί και πλέον είχε έναν πολύ σημαντικό ρόλο σε μια εφαρμογή. Ταυτόχρονα πλαίσια εργασίας στον εξυπηρετητή (server-side frameworks) επέτρεψαν τον διαμοιρασμό της λογικής μεταξύ πελάτη – εξυπηρετητή. Πλέον ο πελάτης δεν αιτείται μόνο HTML περιεχόμενο αλλά μπορεί να ζητήσει και δεδομένα από το backend, και όταν αυτά επιστραφούν να τα απεικονίσει / επεξεργαστεί όπως απαιτείται στην κάθε περίσταση. Με την έλευση της HTML5, οι browsers ισχυροποιήθηκαν και προστέθηκαν πολλές δυνατότητες. Σταδιακά η προσοχή του δικτυακού προγραμματισμού στράφηκε προς τις RIAs (Rich Internet Application) , εφαρμογές, οι οποίες έχουν πολλά από τα χαρακτηριστικά των native, αλλά συνήθως η εκτέλεσή τους απαιτεί την εγκατάσταση κάποιας επέκτασης στο browser. Η επιστημονική κοινότητα αντιλήφθηκε αυτή τη νέα δυναμική που αναπτυσσόταν και πλέον από το 2012, σύμφωνα με τις τάσεις που καταγράφει η Google [15] [16] άρχισε σταδιακά να εγκαταλείπει τα frameworks που στηρίζονταν σε plug-ins και third-party εργαλεία. Τη μεταστροφή αυτή βοήθησε και η αντικατάσταση των τελευταίων με πληθώρα νέων που αξιοποιούσαν τα βελτιωμένα APIs που προσέφεραν πλέον οι browsers και την επεξεργαστική ισχύ των προσωπικών υπολογιστών και διευκόλυναν την ανάπτυξη και συντήρηση μεγάλης έκτασης εφαρμογών.

Αυτό σηματοδοτεί την εποχή όπου τα προγράμματα πελάτες (Client Programs) σταματούν να είναι thin clients (μικρός φόρτος εργασίας, ελάχιστες αρμοδιότητες, απλές δυνατότητες αλληλεπίδρασης χρήστη – εφαρμογής, εμφάνιση περιεχομένου) και πλέον γίνονται thick clients. Έτσι τώρα τα προγράμματα – πελάτες (ή αλλιώς frontend) αναλαμβάνουν υπολογιστικά περίπλοκες εργασίες, αλληλεπιδρούν άμεσα με το backend και η λογική παρουσίασης μετατοπίζεται στον frontend/πελάτη.



Εικόνα 3-8: : Διαφορά κύκλου ζωής μεταξύ της υπάρχουσας λογικής του διαδικτύου και της SPA

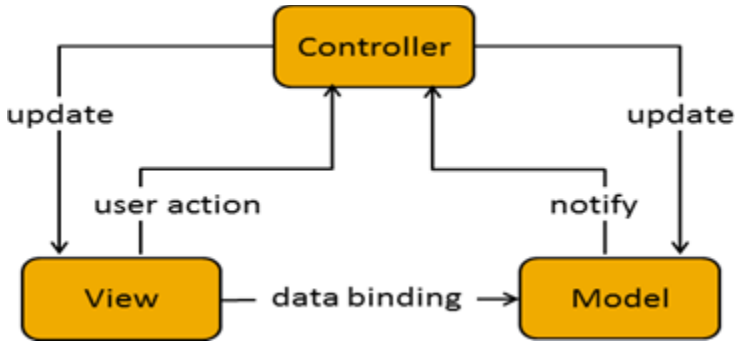
3.1.5.3 SPA και το Μοντέλο MVC

Η επόμενη μεγάλη ιδέα ήρθε το 2002 με την παρουσίαση της ιδέας των εφαρμογών μονής σελίδας (Single Page Applications). Η λογική εδώ είναι ότι μια web εφαρμογή λαμβάνει χώρα σε μία μόνο ιστοσελίδα με στόχο να παρέχει μια πιο άμεση και ομαλή εμπειρία συγκρίσιμη με εφαρμογές επιφάνειας εργασίας (desktop applications). Η σελίδα φορτώνεται μία μόνο φορά και στη συνέχεια τα δεδομένα φορτώνονται δυναμικά μέσω αλληλεπίδρασης server – client, ενώ ταυτόχρονα εξακολουθεί να δίνει στον χρήστη την εντύπωση ότι πλοηγείται σε διαφορετικές σελίδες, μέρη της εφαρμογής.

Η πολυπλοκότητα και η ποσότητα της λογικής που δόθηκε στον πελάτη δημιούργησε νέα προβλήματα: το μέγεθος του κώδικα αυξήθηκε δραματικά, και οι τεχνοτροπίες που δημιουργήθηκαν ποίκιλλαν πολύ. Η λύση δόθηκε ομαδοποιώντας πρακτικές και σχεδιαστικές επιλογές σε ολοκληρωμένες τεχνοτροπίες (frameworks) που

μέχρι στιγμής υπήρχαν μόνο στους εξυπηρετητές. Η πιο γνωστή τεχνοτροπία όλων αυτών των JavaScript Frameworks είναι η Model – View – Controller (MVC pattern) η οποία αποτελείται από τα εξής τρία μέρη:

- Μοντέλα (Models) που αναπαριστούν τις σχετικές με την εφαρμογή πληροφορίες και δεδομένα, όπως συγκεκριμένες κλάσεις-δοχεία (container classes) δεδομένων. Τα μοντέλα μπορούν να ενημερώνουν τυχόν παρατηρητές όταν η κατάστασή τους αλλάζει (π.χ. όταν κάποια πληροφορία που κρατούν ενημερώνεται ή διαγράφεται).
- Όψεις (Views) που τυπικά θεωρούνται ως η διεπαφή του χρήστη με την εφαρμογή (π.χ. ο κώδικας HTML και CSS). Πρέπει να γνωρίζουν για την ύπαρξη των μοντέλων, έτσι ώστε να τα παρατηρούν, αλλά δεν επικοινωνούν κατευθείαν μαζί τους.
- Ελεγκτές (Controllers) που υλοποιούν τη λογική παρουσίασης (presentation logic) της εφαρμογής. Αυτοί είναι που παίρνουν τις αποφάσεις και ο συνδετικός κρίκος μεταξύ Μοντέλων και Όψεων.



Εικόνα 3-9: Συνήθεις αλληλεπιδράσεις των MVC συστατικών

Η πληθώρα των MVC frameworks που υπάρχουν διαθέσιμα σήμερα, ανάλογα με τους στόχους τους, δεν υλοποιούν με τον ίδιο τρόπο το παραπάνω προγραμματιστικό μοτίβο αλλά συνήθως κάποια παραλλαγή του, με σκοπό πάντα όμως την επιβολή της σχεδιαστικής αρχής γνωστής ως SoC (Separation of Concerns). Παρατηρείται, για παράδειγμα, η συγχώνευση του ρόλου του ελεγκτή και της όψης σε μία οντότητα με αυξημένες αρμοδιότητες και λειτουργικότητα. Γι' αυτό το λόγο, οι υλοποιήσεις που στηρίζονται σε τέτοιου είδους πιο «χαλαρές» προσεγγίσεις συναντώνται στη βιβλιογραφία και ως MV.

3.1.6 Γραφικά για εφαρμογές ιστού

Η ανάπτυξη μιας διαδραστικής εφαρμογής βασισμένης σε browser με σκοπό την απεικόνισης δεδομένων ιατρικής εικόνας, προϋποθέτει την εξέταση των διαθέσιμων εργαλείων για τη δημιουργία δισδιάστατων και τρισδιάστατων γραφικών μέσα στο browser. Γενικά, στο παρελθόν διάφορες μέθοδοι παρέχονταν για το σκοπό αυτό από τους δημιουργούς τους αλλά γεγονός είναι πως διέφεραν από υλοποίηση σε υλοποίηση. Μόλις πρόσφατα, αρχικά με την εισαγωγή της HTML5 και αργότερα του WebGL, αναδείχθηκαν κάποια πρότυπα που έτυχαν ευρείας συμμόρφωσης.

Η πέμπτη έκδοση της HTML, μεταξύ των άλλων, εισήγε και μια σειρά από σημαντικά χαρακτηριστικά που σχεδιάστηκαν για τη διευκόλυνση του χειρισμού πολυμέσων και γραφικού περιεχομένου στον ιστό χωρίς να απαιτείται η χρήση «κλειστών» επεκτάσεων (proprietary plugins) και APIs. Ένα από αυτά τα νέα χαρακτηριστικά είναι το στοιχείο <canvas>, ένα «προγραμματιζόμενο» στοιχείο γραφικής απεικόνισης που στην ουσία αποτελεί ένα χαμηλού επιπέδου διαδικαστικό μοντέλο το οποίο ενημερώνει ένα bitmap. Αρχικά παρουσιάστηκε από την Apple [17]. Το συστατικό αυτό συνιστά τη βάση για τις πιο περίπλοκες τεχνικές απεικόνισης δισδιάστατου και τρισδιάστατου περιεχομένου στους σύγχρονους browsers.

3.1.7 Δισδιάστατα γραφικά

Παραδοσιακά, η απεικόνιση δισδιάστατων γραφικών αποτελούσε αναπόσπαστο κομμάτι των περισσοτέρων browsers. Εκτός από την απλή εμφάνιση αρχείων εικόνας (JPEG, PNG, BMP, κτλ.), υπήρχε μια ποικιλία από προσεγγίσεις και μεθόδους που επέτρεπαν την παραγωγή και το χειρισμό δισδιάστατου γενικά περιεχομένου. Η CSS, όπως παρουσιάστηκε παραπάνω, χρησιμοποιείται για να αλλάξει τον τρόπο εμφάνισης ενός HTML στοιχείου, αλλά δεν κρίνεται πραγματικά κατάλληλη για τη δημιουργία περίπλοκων γραφικών λόγω απουσίας εντολών σχεδίασης. Το κενό αυτό προσπάθησε να καλύψει ήδη από το 1999 ο οργανισμός W3C όταν ξεκίνησε να προδιαγράφει ένα νέο πρότυπο εικόνων, το SVG (Scalable Vector Graphics) [18]. Παραστατικά, το SVG μπορεί να θεωρηθεί ως το αντίστοιχο της HTML για τα γραφικά του υπολογιστή. Είναι μια γλώσσα σήμανσης εγγράφου (markup language) για την περιγραφή όλων των πτυχών μιας εικόνας, από τη γεωμετρία των σχημάτων, στο στυλ του κειμένου, στην κίνηση και την παρουσίαση πολυμέσων, όπως βίντεο και ήχου. Είναι πλήρως διαδραστική, και περιλαμβάνει επίσης τη δική της έννοια του προγραμματιζόμενου DOM. Υποστηρίζει ένα μεγάλο εύρος από οπτικά χαρακτηριστικά όπως χρωματική διαβάθμιση (gradient), αδιαφάνεια (opacity), φίλτρα, κ.ά. Η χρήση εικόνων SVG επιτρέπει πλήρως κλιμακούμενα (scalable), ομαλά, επαναχρησιμοποιούμενα γραφικά για μια ποικιλία περιπτώσεων (παιχνίδια, επιστημονικές

απεικονίσεις, κτλ.). Υποστηρίζεται δε από τους περισσότερους σημερινούς browsers, κινητές συσκευές και set-top boxes. Μπορούν να παραχθούν από γλώσσες για client-side ή server-side προγραμματισμό, αλλά συνήθως προτιμάται εξειδικευμένο λογισμικό που έχει εξελιχθεί για την απλοποίηση της όλης διαδικασίας.

Τελικά, η πιο πρόσφατη εναλλακτική ήρθε με την άφιξη της HTML5. Το Canvas είναι ένα προγραμματιζόμενο γραφικό στοιχείο, υποστηριζόμενο από όλους τους μεγάλους browsers, που μπορεί να ελεγχθεί με JavaScript, και αποτελεί σήμερα τη πιο δημοφιλή επιλογή για απεικόνιση δισδιάστατου υλικού στο διαδίκτυο. Στην ουσία, πρόκειται για μια ορθογώνια περιοχή της HTML σελίδας που μπορεί να ζωγραφιστεί με χρήση ενσωματωμένων μεθόδων για το σκοπό αυτό. Είναι ένα αρκετά χαμηλού επιπέδου μοντέλο χωρίς να διαθέτει αντίληψη περί γράφων σκηνής (scene graphs) ή παρουσία DOM και για το λόγο αυτό είναι ιδιαίτερα αποδοτικό. Προγραμματιστές, λοιπόν, είναι σε θέση να εξελίσσουν παιχνίδια ή ακόμη και εφαρμογές πλήρους χαρακτηριστικών χρησιμοποιώντας το Canvas API μόνο ή σε συνδυασμό με SVG. Να σημειωθεί πως πριν την έλευση του Canvas, αντίστοιχη λειτουργικότητα παρείχαν στους browsers κάποιες επεκτάσεις που απαιτούσαν ξεχωριστή εγκατάσταση. Οι πιο γνωστές εκ των οποίων είναι ο Flash Player της Adobe (συνεχίζει να υποστηρίζεται από κάποιες συσκευές) και τα Java applets (θεωρούνται πλέον παρωχημένα).

3.1.8 Τρισδιάστατα γραφικά

Τα τρισδιάστατα γραφικά συνήθως ορίζονται από ένα χώρο καρτεσιανών συντεταγμένων στον οποίο βρίσκονται τοποθετημένα τρισδιάστατα αντικείμενα, καθώς επίσης και ένα αντικείμενο που λειτουργεί ως κάμερα μέσα από την οποία η όλη σκηνή προβάλλεται με τη βοήθεια ενός πίνακα προβολής (projection matrix). Τα τρισδιάστατα γραφικά είναι γενικά περισσότερο απαιτητικά από άποψη απαιτούμενων υπολογιστικών πόρων σε σχέση με τα δισδιάστατα λόγω των πιο περίπλοκων μαθηματικών υπολογισμών που περιλαμβάνουν. Επιπρόσθετα, η απεικόνιση μιας τρισδιάστατης σκηνής (3D rendering) σημαίνει πως η εικόνα πρέπει ιδανικά να ανανεώνεται με ρυθμό 60 καρέ το δευτερόλεπτο, στόχος που αποτελεί μεγάλη πρόκληση για τους προγραμματιστές.

3.1.8.1 Προ-WebGL

Γενικά μιλώντας, η ιστορία των τρισδιάστατων γραφικών στο διαδίκτυο μπορεί να διαχωριστεί στην εποχή πριν και μετά την προτυποποίηση του WebGL. Πριν την έλευση του τελευταίου, ο κανονικός τρόπος για την απόδοση τρισδιάστατου περιεχομένου στο browser εξαρτιόταν άμεσα από ειδικές για το σκοπό αυτό επεκτάσεις που ο χρήστης

καλείτο να «κατεβάσει» και να εγκαταστήσει στο μηχάνημά του. Ο Flash Player, που ήδη έχει αναφερθεί για τη χρήση του στα δισδιάστατα γραφικά, ήταν μια από τις κυρίαρχες επεκτάσεις και στο χώρο αυτό. Εντούτοις, η ύπαρξη των επεκτάσεων αυτών δεν ήταν πανάκεια. Χαρακτηριστικό είναι το γεγονός ότι η κάθε μία τους έπρεπε να έχει διαφορετικές υλοποιήσεις για να μπορεί να εκτελείται πάνω από διαφορετικές πλατφόρμες – Windows, Linux, MacOS. Επιπρόσθετα, η Apple από κάποια στιγμή και μετά, αρνήθηκε να υποστηρίξει αυτή τη νέα τάση με τις επεκτάσεις και, αντί αυτού, να επενδύσει στα ανερχόμενα πρότυπα της HTML5. Η εικόνα αυτή, λοιπόν, στην κοινότητα των browsers έδειχνε πως απαιτούσε μια πιο γενική και κοινώς αποδεκτή στο ζήτημα αυτό.

Διάφορες άλλες επεκτάσεις δημιουργήθηκαν, εν τω μεταξύ, προς την κατεύθυνση της εισαγωγής τρισδιάστατου περιεχομένου στο διαδίκτυο. Άλλα χαρακτηριστικά παραδείγματα αποτελούν τα Java applets, είδος μικροεφαρμογών γραμμένων σε Java, που εκτελούνται στο μηχάνημα του χρήστη μέσα σε εικονική μηχανή (απαιτούσαν προφανώς να είναι εγκατεστημένη και κάποια έκδοση της Java). Η Microsoft δημιούργησε τη δική της «κλειστή» υλοποίηση, γνωστή ως Silverlight, ενώ η Google ξεκίνησε την εξέλιξη της δικής της εκδοχής, γνωστής ως O3D[4].

3.1.8.2 WebGL

Παράλληλα με την προτυποποίηση της HTML5, το WebGL έγινε ευρύτερα γνωστό και υποστηριζόμενο, βοηθώντας στον περιορισμό της πληθώρας των επεκτάσεων που χρησιμοποιούνταν από τους προγραμματιστές μέχρι τότε και των ασυμβατοτήτων που η χρήση τους εισήγε. Γενικά, το WebGL επιτρέπει στους προγραμματιστές εισάγουν διαδραστικά τρισδιάστατα γραφικά στο browser σε πραγματικό χρόνο. Μπορεί να εφαρμοστεί σε διαδραστικά μουσικά βίντεο, παιχνίδια, οπτικοποίηση δεδομένων, μορφές τέχνης, τρισδιάστατα περιβάλλοντα σχεδίασης, τρισδιάστατη μοντελοποίηση αντικειμένων, σχεδιασμό μαθηματικών συναρτήσεων, δημιουργία φυσικών προσομοιώσεων, κ.ά. Ήδη χρησιμοποιείται μεταξύ άλλων σε εφαρμογές όπως το Google Maps, το Autocad Cloud της Autodesk και το Citadel demo της Epic Games.

Το WebGL ελέγχεται αποκλειστικά μέσω JavaScript και «τρέχει» σε ένα συγκεκριμένο πλαίσιο (context) του HTML <canvas> στοιχείου, που δίνει πρόσβαση σε τρισδιάστατη επιτάχυνση υλικού. Προτυποποιήθηκε από το Khronos group [19], ενεργά μέλη του οποίου μεταξύ άλλων είναι όλοι οι μεγάλοι παίκτες στην αγορά των browsers, εξαιρουμένης της Microsoft. Είναι βασισμένο πάνω στο OpenGL ES 2.0, την έκδοση του ευρέως χρησιμοποιούμενου προτύπου για παραγωγή τρισδιάστατων γραφικών σε εφαρμογές

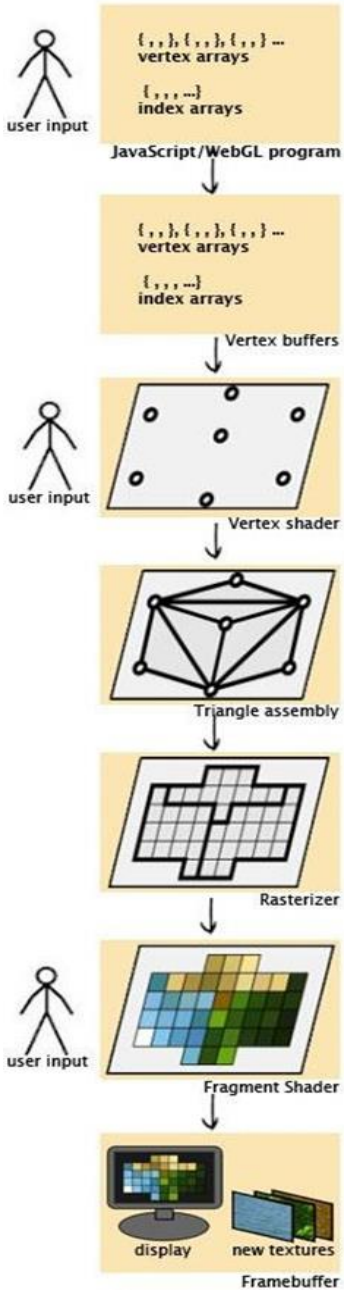
[4] Ενώ αρχικά η υλοποίησή της βασιζόταν σε μια plug-in based αρχιτεκτονική και χρησιμοποιούσε τη C, το Μάη του 2010, η Google ανακοίνωσε πως το όλο project αλλάζει προσανατολισμό και από plug-in θα γίνει port σε βιβλιοθήκη της JavaScript που θα τρέχει πάνω από WebGL [50].

προσωπικών υπολογιστών OpenGL, προσανατολισμένου σε ενσωματωμένα συστήματα [20]. Η επιλογή αυτή έγινε για λόγους συνοχής και συμβατότητας μεταξύ διαφορετικών πλατφόρμων και συσκευών – ως API θα πρέπει να είναι προσβάσιμο από browsers εγκατεστημένους τόσο σε φορητές συσκευές, όπως smartphones και tablets, όσο και προσωπικούς υπολογιστές ανεξαρτήτως λειτουργικού συστήματος [21].

Γεγονός είναι πως το WebGL είναι αρκετά πιο πολύπλοκο από τις υπόλοιπες τεχνολογίες των browsers επειδή σχεδιάστηκε για την απευθείας «συνεργασία» με την κάρτα γραφικών (GPU). Ως αποτέλεσμα, είναι ένα χαμηλού επιπέδου (low-level) API, η χρήση του οποίου όμως, ως αντιστάθμισμα, επιτυγχάνει υψηλές επιδόσεις κατά τον τρισδιάστατο σχεδιασμό (rendering).

Ο προγραμματισμός με το WebGL στοχεύει συνήθως στην απεικόνιση κάποιου είδους σκηνής που απαρτίζεται από μοντέλα. Αυτή εμπεριέχει πολλαπλές διαδοχικές κλήσεις σχεδίασης (draw calls), κάθε μία από τις οποίες εκτελείται στην κάρτα γραφικών μέσω μιας διαδικασίας γνωστής ως διασωλήνωση απεικόνισης (rendering pipeline, βλ. Εικόνα 3-10). Δομικό στοιχείο όλων των μοντέλων που σχεδιάζονται είναι το τρίγωνο, οπότε η μέθοδος σχεδιασμού περιέχει τη δημιουργία πληροφορίας για το πού αυτά τα τρίγωνα θα σχεδιαστούν και το πώς θα φαίνονται (χρώμα, υφή, σκιά, κτλ.). Η πληροφορία αυτή τότε παρέχεται στη GPU, που την επεξεργάζεται και έπειτα επιστρέφει μια όψη της σκηνής.

Η διαδικασία ξεκινά με τη δημιουργία πινάκων κορυφών (vertex arrays), που περιέχουν γνωρίσματα που περιγράφουν κορυφές όπως η τοποθεσία τους, το χρώμα, η υφή, η επίδραση του φωτός (vertex normal) πάνω τους, κ.ά. Οι πίνακες αυτοί κατασκευάζονται σε JavaScript και, λόγω αυξημένης πολυπλοκότητας, συνήθως χρησιμοποιείται κάποια βιβλιοθήκη που παρέχει έτοιμους για διάφορα γεωμετρικά σχήματα.



Εικόνα 3-10: Η διασωλήνωση απεικόνισης (rendering pipeline) του WebGL

Τα δεδομένα τότε των vertex arrays αντιγράφονται σε ειδικούς για το σκοπό αυτό vertex buffers της GPU. Όταν κατατίθεται μια εργασία σχεδίασης, στη GPU πρέπει επίσης να παρασχεθεί ένας επιπλέον πίνακας (index array) τα στοιχεία του οποίου δεικτοδοτούν τα στοιχεία του πίνακα κορυφών. Χρησιμοποιείται για το μετέπειτα έλεγχο της συγκέντρωσης των κορυφών ώστε να σχηματίσουν τρίγωνα.

Η GPU ξεκινά διαβάζοντας κάθε κορυφή από το vertex buffer και περνώντας την σε ένα vertex shader. Vertex shader ονομάζεται ένα πρόγραμμα – γραμμένο σε GLSL για να εκτελείται παράλληλα από τις σύγχρονες GPUs [22] – το οποίο δέχεται κάθε φορά σαν όρισμα τα γνωρίσματα μιας κορυφής που πρόκειται να επεξεργαστεί. Κατ'ελάχιστο, ο vertex shader υπολογίζει την προβολή της κορυφής στο επίπεδο της οθόνης αλλά μπορεί να παράγει και άλλες πληροφορίες όπως χρώμα ή συντεταγμένες υφής (texture coordinates) για κάθε κορυφή.

Η GPU ύστερα «συνδέει» τις προβολές των κορυφών που έχουν προκύψει από το vertex shader για να σχηματίσει τρίγωνα. Το καταφέρνει αυτό παίρνοντας τις κορυφές με τη σειρά που υποδεικνύει ο index array και ομαδοποιώντας τες σε σύνολα των τριών έκαστο.

Αμέσως μετά ο rasterizer παίρνει κάθε τρίγωνο, το ψαλιδίζει, απορρίπτει τα μέρη που θα μείνουν εκτός οθόνης και «σπάει» τα υπόλοιπα ορατά σε τεμάχια μεγέθους pixel (pixel-sized fragments). Στην υπόλοιπη – χρωματική κυρίως – πληροφορία που προκύπτει από την επεξεργασία του vertex shader εφαρμόζεται παρεμβολή (interpolation) καθ' όλη την έκταση της επιφάνειας του κάθε τριγώνου, αναθέτοντας μία ομαλή χρωματική διαβάθμιση σε κάθε fragment. Για παράδειγμα, αν ο vertex shader εκχωρήσει μία χρωματική τιμή σε κάθε κορυφή, ο rasterizer θα πραγματοποιήσει κατάλληλη μίξη των χρωμάτων των γειτονικών κορυφών, ώστε η επιφάνεια να έχει όσο γίνεται περισσότερο απαλές χρωματικές διακυμάνσεις.

Τα παραγόμενα pixel-sized fragments τροφοδοτούνται κατόπιν σε ένα ακόμα πρόγραμμα που ονομάζεται fragment shader. Ο fragment shader γράφεται και αυτός από το προγραμματιστή σε GLSL, εκτελείται παράλληλα από τις GPUs και εξάγει τιμές χρώματος και βάθους για κάθε pixel, οι οποίες γράφονται στη συνέχεια στο frame-buffer. Συνήθεις λειτουργίες που εκτελεί είναι η αντιστοίχιση υφής (texture mapping) και προσομοίωση φωτισμού. Αφού ο shader εκτελείται ανεξάρτητα για κάθε pixel του Canvas, μπορεί να χρησιμοποιηθεί για να παράγει τα πιο πολύπλοκα και περίτεχνα οπτικά εφέ, ωστόσο πρέπει πάντα να λαμβάνεται υπόψιν πως είναι και το ευαίσθητο κομμάτι της διασωλήνωσης από άποψη επίδοσης.

Ο framebuffer είναι ο τελικός προορισμός της διαδικασίας σχεδίασης που εκτελεί η GPU. Ο framebuffer δεν αποτελεί απλώς μια απλή δισδιάστατη εικόνα: εκτός από έναν η περισσότερους επιμέρους χρωματικούς buffers μπορεί επιπλέον να έχει και κάποιον depth buffer ή/και stencil buffer. Και οι δύο τελευταίοι αυτοί buffers χρησιμοποιούνται για να «φιλτράρουν» τα fragments πριν αυτά αποδοθούν στο framebuffer. Ο έλεγχος βάθους απορρίπτει fragments αντικειμένων που βρίσκονται πίσω από ήδη σχεδιασμένα και ο έλεγχος με στένσιλ χρησιμοποιεί σχήματα «ζωγραφισμένα» στο stencil buffer για να περιορίσει το μέρος της εικόνας που τελικά θα αποδοθεί. Τα fragments εκείνα που δεν απορρίφθηκαν από το φιλτράρισμα του τελευταίου αυτού σταδίου αποδίδονται τελικώς στην οθόνη του χρήστη.

Η εξοικείωση με το WebGL API, όπως γίνεται κατανοητό, είναι χρονοβόρα και η απευθείας χρήση του μια επίπονη διαδικασία, ιδιαίτερα για προγραμματιστές χωρίς μεγάλη εμπειρία από τρισδιάστατη απεικόνιση. Για το σκοπό αυτό, έχουν αναπτυχθεί πολλές αξιόλογες βιβλιοθήκες που προσθέτουν μια αφαιρετική στρώση πάνω από τις χαμηλού επιπέδου κλήσεις του WebGL, επιταχύνοντας και απλοποιώντας την ανάπτυξη εφαρμογών. Η πιο διαδεδομένη από αυτές, η οποία επιλέχθηκε και για τη συγκεκριμένη υλοποίηση ονομάζεται Three.js (βλ. Ενότητα 3.2.3).

3.1.9 WebRTC

3.1.9.1 Σύντομη ιστορική προσέγγιση

Μία από τις τελευταίες μεγάλες προκλήσεις για το διαδίκτυο είναι η παροχή της δυνατότητας για ανθρώπινη επικοινωνία μέσω ήχου και βίντεο, κάτι που επικράτησε να λέγεται επικοινωνία πραγματικού χρόνου (Real-Time Communication - RTC). Οι οραματιστές της κοινότητας θεωρούσαν ανέκαθεν πως οι RTC δυνατότητες θα πρέπει να είναι εξίσου φυσικές για τις εφαρμογές ιστού, όπως είναι και η εισαγωγή κειμένου σε φόρμες. Χωρίς αυτές αισθάνονταν περιορισμό ως προς την ικανότητά τους να καινοτομούν και να εξελίσσουν νέους τρόπους για την αλληλεπίδραση των χρηστών.

Εντούτοις, ιστορικά η επικοινωνία πραγματικού χρόνου επιτυγχανόταν μέσα σε επιχειρησιακά περιβάλλοντα, καθώς, λόγο αυξημένης πολυπλοκότητας, απαιτούσε την απόκτηση ακριβών και αδειοδοτημένων τεχνολογιών βίντεο και ήχου ή την εξέλιξή τους από την ίδια την επιχείρηση. Επιπρόσθετα, η ενοποίηση των RTC τεχνολογιών με το ήδη υπάρχον περιεχόμενο, δεδομένα και υπηρεσίες υπήρξε δύσκολο και χρονοβόρο έργο, ειδικά στον ιστό.

Το Gmail video chat έγινε δημοφιλές το 2008 και το 2011 η Google εισήγαγε την εφαρμογή Hangouts, η οποία χρησιμοποιούσε την ίδια υπηρεσία με το Gmail, γνωστής ως Google Talk. Στη συνέχεια, η ίδια εταιρία αγόρασε τη GIPS. Η τελευταία είχε εξελίξει μια σειρά από συστατικά (component) απαραίτητα για RTC, όπως κωδικοποιητές και τεχνικές ακύρωσης ηχούς (echo cancellation techniques), τα οποία κατόπιν κατέστησε «ανοικτά» στο κοινό (open source). Συνεργάστηκε επίσης με τις αρμόδιες επιτροπές IETF [23] και W3C [24] για να επιτύχει την αποδοχή της βιομηχανίας. Το Μάη του 2011, η Ericsson κατασκεύασε την πρώτη υλοποίηση του WebRTC (Web Real-Time Communication).

Το WebRTC περιέχει σήμερα ανοικτά πρότυπα για πραγματικού χρόνου επικοινωνία μέσω βίντεο, ήχου και δεδομένων στο browser, χωρίς την ανάγκη επεκτάσεων. Η ανάγκη για ένα τέτοιο σύνολο προτύπων είναι πράγματι υπαρκτή:

- Πολλές υπηρεσίες ιστού (web services) ήδη χρησιμοποιούν RTC τεχνικές, αλλά απαιτούν τη λήψη και εγκατάσταση native εφαρμογών ή επεκτάσεων (αν αναφερόμαστε στο browser). Παραδείγματα τέτοιων δημοφιλών εφαρμογών αποτελούν το Skype, το Facebook (το οποίο χρησιμοποιούσε την υποδομή του Skype μέχρι πρότινος που αποφάσισε να υιοθετήσει το WebRTC) και το Google Hangouts (που χρησιμοποιεί τη Google Talk επέκταση).
- Η λήψη, εγκατάσταση και ενημέρωση των native εφαρμογών και των επεκτάσεων μπορεί να είναι περίπλοκη, επιρρεπής σε σφάλματα και ενοχλητική ακόμα για τον απλό χρήστη.
- Ειδικά οι επεκτάσεις μπορεί να είναι απαιτητικές για την ανάπτυξη, την αποσφαλμάτωση, την επιδιόρθωση, τον έλεγχο και τη συντήρησή τους – μπορεί ακόμα να απαιτείται η αδειοδότηση και η ενοποίηση με πολύπλοκη και ακριβή τεχνολογία. Επίσης, συχνά είναι δύσκολο ακόμα και να πειστεί αρχικά ο χρήστης να τις εγκαταστήσει και να τις χρησιμοποιήσει.

Σύμφωνα με τις κατευθυντήριες αρχές που συνοδεύουν το WebRTC project, τα APIs του πρέπει να είναι ανοικτού κώδικα, ελεύθερα, προτυποποιημένα, ενσωματωμένα στους browsers και σαφώς πιο αποδοτικά από τις ήδη υπάρχουσες τεχνολογίες.

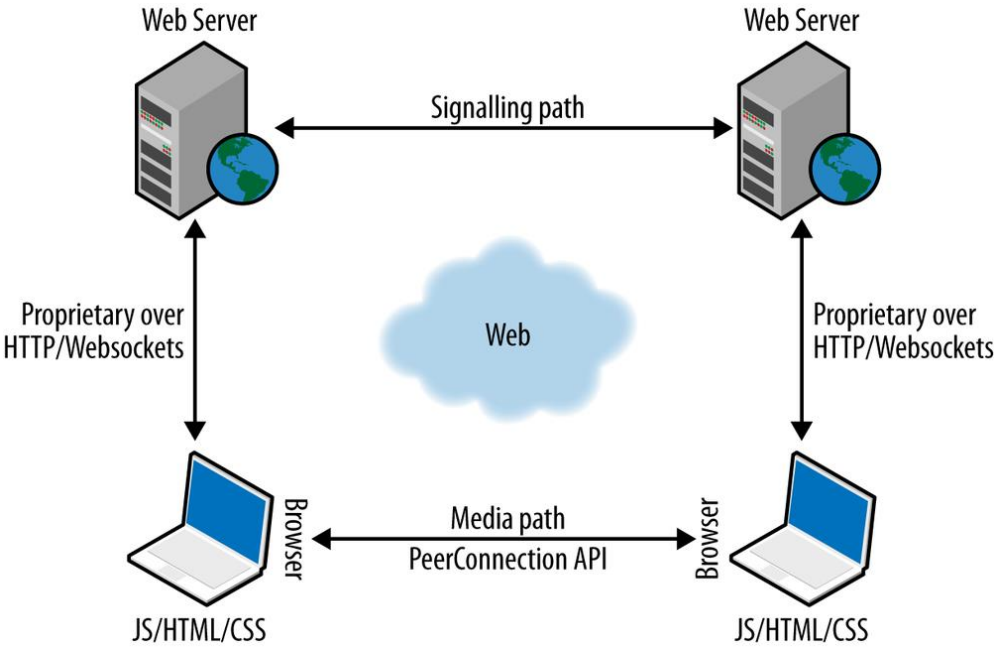
3.1.9.2 Το WebRTC ως τεχνολογία

Με τον όρο WebRTC καλείται ένα σύνολο από προδιαγραφές, πρωτόκολλα και JavaScript APIs, ο συνδυασμός των οποίων επιτρέπει τον peer-to-peer διαμοιρασμό ήχου, βίντεο και δεδομένων μεταξύ browsers (peers). Αντί να βασίζεται σε εξωτερικές επεκτάσεις (third-party plug-ins) ή κλειστό (proprietary) λογισμικό, το WebRTC μετατρέπει την επικοινωνία πραγματικού χρόνου σε ένα καθιερωμένο χαρακτηριστικό στο οποίο μπορεί να έχει πρόσβαση οποιαδήποτε εφαρμογή ιστού με τη μορφή τριών ακόμα απλών και διαισθητικών JavaScript APIs.

Το πρώτο από αυτά ονομάζεται MediaStream (ευρύτερα γνωστό και ως getUser-Media) και αναπαριστά συγχρονισμένες ροές πολυμέσων που μπορούν να μεταδοθούν μεταξύ των peers. Μια τέτοια ροή (stream) μπορεί να αποτελείται από ένα ή περισσότερα κομμάτια, αν και συνήθως αποτελείται από ένα για το βίντεο και ένα για τον ήχο. Μια ροή μπορεί να μεταδώσει είτε «ζωντανά» πολυμέσα (για ηχητικές κλήσεις ή βιντεοδιασκέψεις) ή αποθηκευμένα (π.χ. ταινίες ή μουσική που βρίσκονται στον τοπικό σκληρό δίσκο). Το δεύτερο, και πιο σημαντικό ίσως, είναι το RTCPeerConnection, υπεύθυνο για τη σταθερή και αποδοτική μετάδοση των ροών δεδομένων μεταξύ των peers. Χάρη σε αυτό, οι προγραμματιστές κατά την ανάπτυξη των εφαρμογών δε χρειάζεται να ασχοληθούν με χαμηλού επιπέδου ζητήματα, όπως διαχείριση χαμένων πακέτων και διαθέσιμου bandwidth, ακύρωση ηχούς, «καθαρισμό» ήχου και βίντεο, κ.ά. Τέλος, εκτός από ήχο και

βίντεο, το WebRTC, όπως προαναφέρθηκε, υποστηρίζει επίσης μετάδοση και άλλων τύπων δεδομένων σε πραγματικό χρόνο. Το σκοπό αυτό εξυπηρετεί το RTCDataChannel API. Επιτρέπει την αποστολή τυχαίων δεδομένων (arbitrary data) με χαμηλή καθυστέρηση (low latency) και υψηλή διεκπεραιωτικότητα. Μπορεί να χρησιμοποιηθεί μεταξύ άλλων για ανάπτυξη παιχνιδιών, εφαρμογών απομακρυσμένης επιφάνειας εργασίας, δωμάτια συνομιλιών, μεταφορά αρχείων μέχρι και επικοινωνία αποκεντροποιημένων δικτύων.

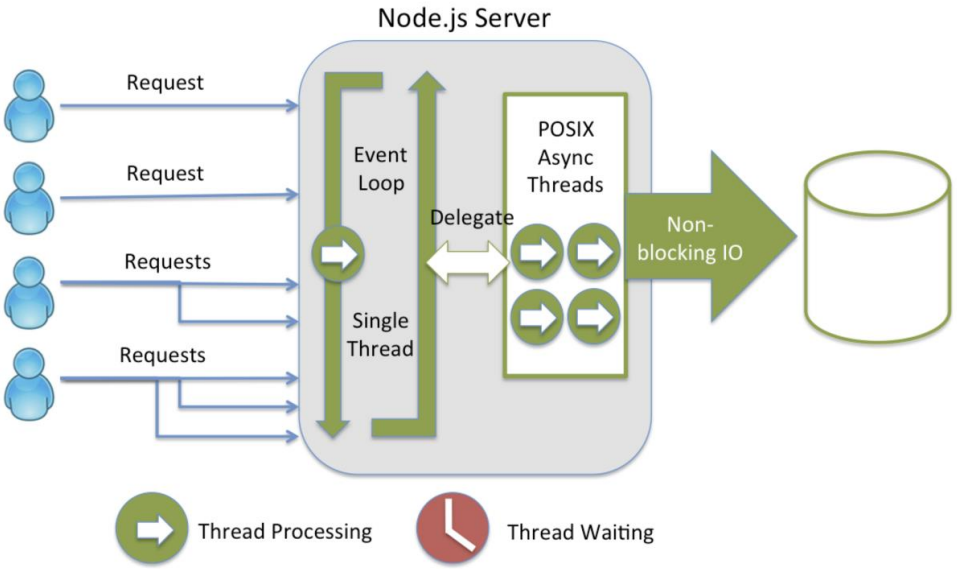
Το WebRTC αξιοποιεί το RTCPeerConnection API για την ανταλλαγή ροών δεδομένων μεταξύ browsers/peers, αλλά όπως είναι εύλογο, απαιτεί ένα μηχανισμό για να συντονίσει την εγκαθίδρυση των απομακρυσμένων συνδέσεων μέσω της αποστολής μηνυμάτων ελέγχου – μια διαδικασία γνωστή ως σηματοδοσία (signaling). Συγκεκριμένες μέθοδοι και πρωτόκολλα δεν παρέχονται για το σκοπό αυτό από το WebRTC. Οι κατασκευαστές λογισμικού αφήνονται ελεύθεροι να επιλέξουν από μια ευρεία γκάμα έτοιμων μηχανισμών και πρωτοκόλλων ανταλλαγής μηνυμάτων (SIP [25], XMPP [26], WebSockets [27]) ανάλογα με τις προσωπικές τους προτιμήσεις. Η ανταλλαγή των απαραίτητων πληροφοριών μέσω σηματοδοσίας πρέπει να έχει ολοκληρωθεί επιτυχώς για να καταστεί δυνατή η μετέπειτα ανταλλαγή οποιασδήποτε μορφής δεδομένων μεταξύ των browsers/peers (Εικόνα 3-11).



Εικόνα 3-11: Κοινή τοπολογία βασισμένη σε WebRTC επικοινωνία

3.1.10 Node.js

Το Node.js είναι μια πλατφόρμα ανάπτυξης λογισμικού κυρίως για δημιουργία εξυπηρετητών (Web Servers) κτισμένη σε περιβάλλον Javascript το οποίο είναι βασισμένο πάνω στη μηχανή JavaScript V8 του Chorme (Chrome's V8 JavaScript engine) και αναπτύχθηκε το 2009 από την εταιρεία Joyuent. Το κύριο μέλημα του δημιουργού του, Ryan Dalh, ήταν να βρεθεί ένας τρόπος, ο χρήστης να ενημερώνεται σε πραγματικό χρόνο για την κατάσταση κάποιου αρχείου που ανέβασε στο διαδίκτυο. Σε αντίθεση με τους παραδοσιακούς server που στηρίζονται στην πολυνηματικότητα, το Node.js ακολουθεί μια διαφορετική προσέγγιση: ασύγχρονη επικοινωνία εισόδου/εξόδου. Όπως και ο κώδικας που τρέχει σε έναν browser βασίζεται στα callbacks και στην ασύγχρονη εκτέλεσή τους. Έτσι, όταν ο επεξεργαστής περιμένει ένα γεγονός, ορίζεται μια συνάρτηση επιστροφής και μέχρι να ολοκληρωθεί μια διαδικασία ο επεξεργαστής μένει ανενεργός και ελεύθερος. Όταν ολοκληρωθεί το γεγονός, καλείται η συνάρτηση αυτή και κρατά τον επεξεργαστή κατειλημμένο μέχρι να ολοκληρωθεί ολόκληρη η διαδικασία. Για παράδειγμα, όταν ο Server λάβει ένα HTTP και πρέπει να διαβάσει ένα αρχείο τότε αναθέτει στο λειτουργικό το άνοιγμα του αρχείου και μέχρι να ολοκληρωθεί η διαδικασία, ξεκινά να εξυπηρετεί το επόμενο request. Όταν ολοκληρωθεί επιτυχώς το άνοιγμα και η ανάγνωση του αρχείου, επιστρέφει το επιθυμητό αποτέλεσμα. Έτσι αυτό που πετυχαίνει είναι η ελαχιστοποίηση της αναμονής και η μικρή απαίτηση σε μνήμη [28] [29].



Εικόνα 3-12: Η λογική εξυπηρέτησης πολλών ταυτόχρονων αιτημάτων

Εκτός από τις πολύ καλές επιδόσεις που παρουσιάζει το Node.js, ένα επιπρόσθετο πλεονέκτημά του είναι η γλώσσα στην οποία είναι υλοποιημένο. Καθώς είναι γραμμένος σε Javascript, οι προγραμματιστές διαδικτυακών εφαρμογών (Web Developer) μπορούν να δημιουργήσουν ολόκληρα τα συστήματα τους σε μία μόνο γλώσσα προγραμματισμού, πράγμα που του επιτρέπει να στρέψει την προσοχή του σε θέματα διαφορετικά από την εκμάθηση διαφορετικών γλωσσών προγραμματισμού. Ένα επιπλέον πλεονέκτημα είναι η ύπαρξη μιας πολύ βολικής και εύχρηστης βιβλιοθήκης, η οποία αποτελείται από μια πληθώρα μικρών αρχείων (modules) που είναι συμβατά με το Node.js και με τη βοήθεια του διαχειριστή αρχείου του Node.js (NPM Node Package Manager) η διαχείριση και εγκατάσταση είναι πολύ εύκολη διαδικασία .

Παρ' όλη την αυξανόμενη δημοτικότητα και επίδοση του, το Node.js έχει κάποια μειονεκτήματα. Δεν επιτρέπει για παράδειγμα επεκτασιμότητα (scalability) καθώς δεν μπορεί να εκμεταλλευτεί τα πολυπύρηνα μηχανήματα της εποχής, ενώ εύκολα δημιουργούνται πολλαπλοί ορισμοί συναρτήσεων επιστροφής, γεγονός το οποίο μπορεί να δημιουργήσει προβλήματα. Επιπλέον το Node.js δεν είναι ικανό για πολύ βαριές υπολογιστικά διαδικασίες αλλά ειδικεύεται σε λειτουργίες Εισόδου/Εξόδου (I/O).

3.2 Frameworks και Βιβλιοθήκες της SPA HERMES

Επειδή όπως αναφέρθηκε η παρούσα υλοποίηση ενσωματώθηκε στη συνέχεια στην Single Page Application HERMES του ΕΔΕΤ, απαραίτητη ήταν η εξοικείωση και η ενσωμάτωση κάποιον διαδεδομένων Framework και βιβλιοθηκών που το HERMES ήδη χρησιμοποιούσε. Το Framework ReactJs χρησιμοποιήθηκε για τη δημιουργία της διεπαφής της εφαρμογής και, σε συνδυασμό με τη βιβλιοθήκη Redux, τη διαχείριση του state αυτής. Σε ότι αφορά το rendering και την απεικόνιση των ιατρικών εικόνων χρησιμοποιήθηκε όπως έχει αναφερθεί η βιβλιοθήκη Three.js, ως ενδιάμεσο στρώμα μεταξύ προγραμματισμού σε υψηλό επίπεδου και του WebGL. Ακολούθως αναφέρονται αυτά καθώς και μία σύντομη περιγραφή τους.

3.2.1 ReactJs

Πρόκειται για ένα JavaScript Framework/library που αποσκοπεί στη δημιουργία διεπαφών χρήστη (UI) εφαρμογών και συντηρείται από το Facebook και μια κοινότητα μεμονωμένων προγραμματιστών και εταιριών, το οποίο έγινε διαθέσιμο τον Μάρτιο του 2013 [30]. Η βιβλιοθήκη απαλλάσσει τον προγραμματιστή από την αυστηρή διαχείριση σε χαμηλό επίπεδο αντικειμένων που υποβάλλονται στο DOM, δίνοντάς του ένα απλούστερο αφαιρετικό μοντέλο και δυνητικά υψηλότερες επιδόσεις. Το ReactJs περιστρέφεται γύρω

από την χρήση δηλωτικής λογικής μέσω αντικειμένων τα οποία ονομάζονται Components (Εξαρτήματα). Κάθε Component αποτελεί ένα μικρό κομμάτι της διεπαφής χρήστη υλοποιώντας κάποιες συγκεκριμένες λειτουργίες. Δίνονται παρακάτω κάποιες βασικές έννοιες απαραίτητες για την κατανόηση του πως λειτουργεί μία εφαρμογή ReactJs:

- **JSX**: Σημαίνει JavaScript XML και είναι μία επέκταση της JavaScript που επιτρέπει την περιγραφή των components που θα συγκροτήσουν την εφαρμογή. Η χρήση JSX είναι προαιρετική καθώς μετά την προ-επεξεργασία της καταλήγει σε κώδικα JavaScript, μπορεί επομένως κάποιος εύκολα να το αποφύγει. Η σύνταξη JSX φαίνεται παρακάτω με ένα απλό παράδειγμα όπου θέλουμε να δηλώσουμε ένα HTML στοιχείο τύπου h1.

```
const element = <Text>Hello, world!</Text>;
```

Εικόνα 3-13: Σύνταξη JSX

Η μεταβλητή element αντιπροσωπεύει ένα component τύπου Text, που χρησιμοποιείται για την προβολή κειμένου [31].

- **Props:** Όλα σχεδόν τα components μπορούν να προσαρμοστούν κατά την δήλωσή τους μέσω παραμέτρων που ονομάζουμε props. Οι παράμετροι αυτοί είναι διαθέσιμοι εντός των components, και μπορούν να χρησιμοποιηθούν για τον έλεγχο της όψης και συμπεριφοράς τους. Ένα component τύπου Image για παράδειγμα χρησιμοποιείται για την προβολή εικόνας, και κάποια από τα κύρια props του είναι το source, που χρησιμοποιείται για τον εντοπισμό της εικόνας που θα προβληθεί, και το style που χρησιμοποιείται για την οπτική προσαρμογή της εικόνας. Τα props ενός component τίθενται κατά την δήλωση του, και δεν μπορούν να μεταβληθούν από κώδικα εντός του component [32].

- **State:** Τα components είναι επίσης stateful, έχουν δηλαδή state (κατάσταση) που συμβολίζεται με ένα JavaScript αντικείμενο. Η κατάσταση ενός component χρησιμοποιείται για τον ίδιο σκοπό που χρησιμοποιούνται τα props, την προσαρμογή του Component. Αντίθετα όμως με τα props, το state τίθεται αρχικά εντός του constructor και μπορεί να μεταβληθεί από τον κώδικα εντός του Component, μέσω μίας μεθόδου που ονομάζεται setState().

Μερικά από τα πλεονεκτήματα της χρήσης ReactJs για την ανάπτυξη εφαρμογών είναι:

- Η χρήση ενός εικονικού DOM που είναι ένα αντικείμενο JavaScript. Αυτό έχει ως επακόλουθο τη βελτίωση της απόδοσης της εφαρμογής, μιας και το εικονικό αυτό DOM είναι πιο γρήγορο από το κανονικό

- Μπορεί να χρησιμοποιηθεί στην πλευρά και του client και του server καθώς και παράλληλα με άλλα framework
- Η χρήση των component και των μοτίβων δεδομένων βελτιώνουν την αναγνωσιμότητα και τη διαφάνεια του κώδικα, τα οποία βοηθάνε στη συντήρηση μεγαλύτερων εφαρμογών

3.2.2 Redux

Το Redux [33] είναι μια βιβλιοθήκη JavaScript ανοιχτού κώδικα για τη διαχείριση της κατάστασης (State) μιας εφαρμογής. Χρησιμοποιείται συνήθως μαζί με βιβλιοθήκες όπως ReactJs ή Angular για το χτίσιμο UIs. Εμπνευσμένη από την αρχιτεκτονική της βιβλιοθήκης Flux του Facebook, φτιάχτηκε από τον Dan Abramov και τον Andrew Clark το 2015 [34].

Η βιβλιοθήκη έχει τη δική της ορολογία, λογική και συμπεριφορά που ο αναγνώστης δεν απαιτείται να γνωρίζει, οπότε κρίθηκε σκόπιμο να γίνει απλώς μια σύντομη αναφορά στις τέσσερεις σημαντικότερες έννοιες που αυτή χρησιμοποιεί:

- **State:** Το State (ή αλλιώς δένδρο state) είναι μια πολύ ευρεία έννοια, αλλά σύμφωνα με το API του Redux συνήθως αναφέρεται στη μοναδική τιμή κατάστασης που διαχειρίζεται από το Store (αναφέρεται παρακάτω) και επιστρέφεται από την κλήση της getState(). Αναπαριστά ολόκληρη την κατάσταση μιας εφαρμογής Redux, που συχνά είναι ένα βαθιά εμφωλευμένο αντικείμενο.

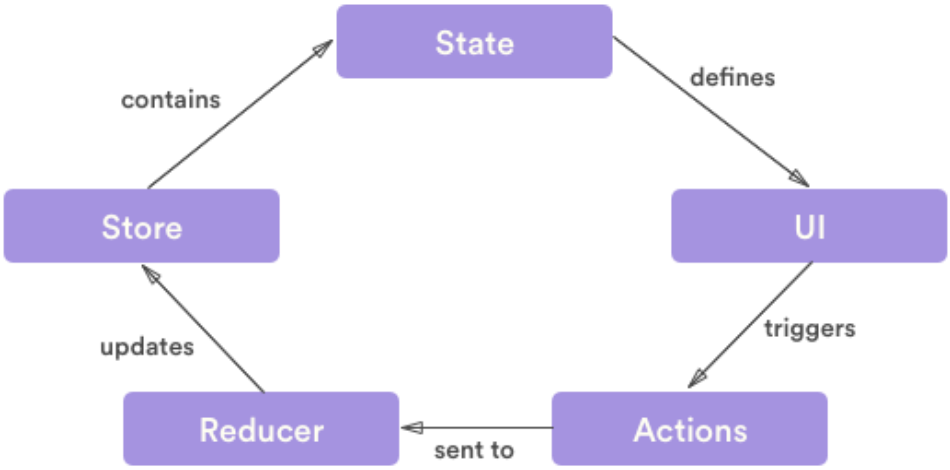
- **Action:** το Action (δράση) είναι ένα απλό αντικείμενο που αντιπροσωπεύει την πρόθεση αλλαγής της κατάστασης. Τα Action είναι ο μόνος τρόπος για να περαστούν τα δεδομένα στο Store. Οποιαδήποτε δεδομένα, είτε προέρχονται από συμβάντα UI, κλήσεις δικτύου είτε άλλες πηγές όπως τα WebSockets, πρέπει τελικά να αποστέλλονται ως Action.

- **Reducer**: Ένας Reducer είναι μια συνάρτηση που δέχεται την τωρινή κατάσταση (State) της εφαρμογής και μια τιμή (από το action) και επιστρέφει ένα νέο State. Το χαρακτηριστικό των Reducer είναι ότι είναι pure συναρτήσεις, δηλαδή για δεδομένες τιμές του τωρινού State και της νέας τιμής εισόδου επιστρέφει δεδομένη έξοδο.

- **Store:** Το Store είναι το object μέσα στο οποίο «ζει» η κατάσταση της εφαρμογής και παρέχει μερικές βοηθητικές μεθόδους για να επιτευχθεί η πρόσβαση στο State, για την αποστολή ενεργειών και για την καταγραφή των listeners. Το σύνολο του State αντιπροσωπεύεται από ένα μόνο Store.

Οποιαδήποτε action επιστρέφει ένα νέο State μέσω των Reducer. Αυτό καθιστά το Redux πολύ απλό και προβλέψιμο.

Στην Εικόνα 3-14 φαίνεται σχηματικά πως αυτά αλληλεπιδρούν.



Εικόνα 3-14: Αλληλεπίδραση μεταξύ των εργαλείων του Redux

3.2.3 Three.js

Αποτελεί μια ευέλικτη και «ελαφριά» τρισδιάστατη μηχανή με μικρή πολυπλοκότητα, που μπορεί να χρησιμοποιηθεί για την απεικόνιση όλων των τεχνολογιών που προαναφέρθηκαν, δηλαδή <canvas>, <svg> και WebGL. Επίσης παρέχει μια σειρά από έτοιμα μοντέλα, εφέ και υλοποιήσεις vertex και fragment shaders που μειώνουν δραστικά την ποσότητα του κώδικα που απαιτείται από τον προγραμματιστή, ενώ η μεγάλη κοινότητα των χρηστών του αποτελεί σημαντική πηγή γνώσεων και συμβουλών [35].

4 Υλοποίηση ανεξάρτητων Demo

4.1 Σκοπός του κεφαλαίου

Η παρούσα διπλωματική εργασία έχει ως σκοπό τη δημιουργία δύο ανεξάρτητων δικτυακών προγραμμάτων προηγμένης επεξεργασίας ιατρικών εικόνων στα πρότυπα μεγάλων DICOM viewer της αγοράς, με σκοπό της απρόσκοπτη και – κυρίως – την αποδοτική λειτουργία τους και μετέπειτα ενσωμάτωση αυτών στην SPA HERMES και τον εμπλουτισμό των δυνατοτήτων της πλατφόρμας ως Medical Imaging εργαλείο. Επομένως, μελετήθηκαν οι παραπάνω λειτουργικότητες σε βάθος και διερευνήθηκε η δυνατότητα υλοποίησης τους σε περιβάλλον Web, με γνώμονα φυσικά την κατάλληλη προσαρμογή τους για τη μετέπειτα ενσωμάτωσή τους στην πλατφόρμα.

Στις επόμενες υποενότητες θα γίνει παρουσίαση των δύο ανεξάρτητων υλοποιήσεων και των προγραμματιστικών και σχεδιαστικών επιλογών που χρειάστηκαν να γίνουν.

4.2 Πολυεπίπεδη Ανασύνθεση – Multiplanar Reconstruction (MPR)

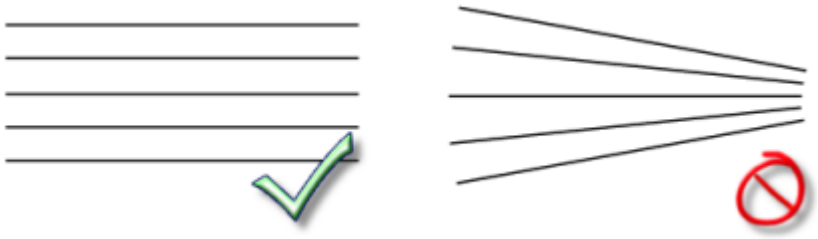
4.2.1 Το MPR στην πράξη

Στο χώρο του Medical Imaging, ως Multiplanar Reconstruction ορίζεται η δυνατότητα ανακατασκευής διαφορετικών δισδιάστατων επιπέδων του χώρου, έχοντας ως είσοδο ένα σετ εικόνων κάποιου συγκεκριμένου επιπέδου. Τα κυριότερα επίπεδα που απεικονίζονται είναι τα εξής τρία κάθετα μεταξύ τους:

- Axial view (αξονικό): μια οριζόντια αποκομμένη φέτα του σώματος όπως φαίνεται από την κορυφή. Συνήθως οι εικόνες DICOM λαμβάνονται σε αυτό το επίπεδο
- Coronal view (στεφανιαίο): μία κατακόρυφη αποκομμένη φέτα του σώματος όπως φαίνεται από εμπρός
- Sagittal view (τοξοειδής): μια κατακόρυφη αποκομμένη φέτα του σώματος όπως φαίνεται από την αριστερή πλευρά του

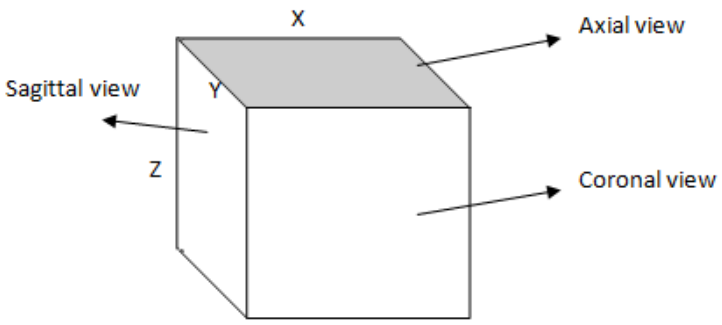
Τα αρχεία DICOM πολλαπλών frame (multi-frame) περιέχουν μια συλλογή από φέτες του σώματος. Έτσι, για παράδειγμα, αν θεωρήσουμε ότι έχουμε ένα multi-frame DICOM αρχείο που περιλαμβάνει F frame (εικόνες) διαστάσεων W x H, τότε το W θα είναι ο άξονας X, το H θα είναι ο άξονας Y και το F θα είναι ο άξονας Z. Είναι επίσης σημαντικό να αναφερθεί ότι η πολυεπίπεδη ανασύνθεση είναι εφικτή όχι μόνο από multi-frame DICOM

αρχεία, αλλά και από μια συλλογή από DICOM αρχεία του ενός frame (single-frame file series) τα οποία όμως θα πρέπει να έχουν το ίδιο πάχος και ύψος μεταξύ τους. Μια άλλη εξίσου σημαντική απαίτηση ώστε να μπορεί να υλοποιηθεί το MPR είναι τα frame να είναι παράλληλα μεταξύ τους.



Εικόνα 4-1: Αριστερά: Παράλληλα μεταξύ τους frame, μπορεί να γίνει MPR. Δεξιά: Συγκλίνοντα frame, δε μπορεί να γίνει MPR

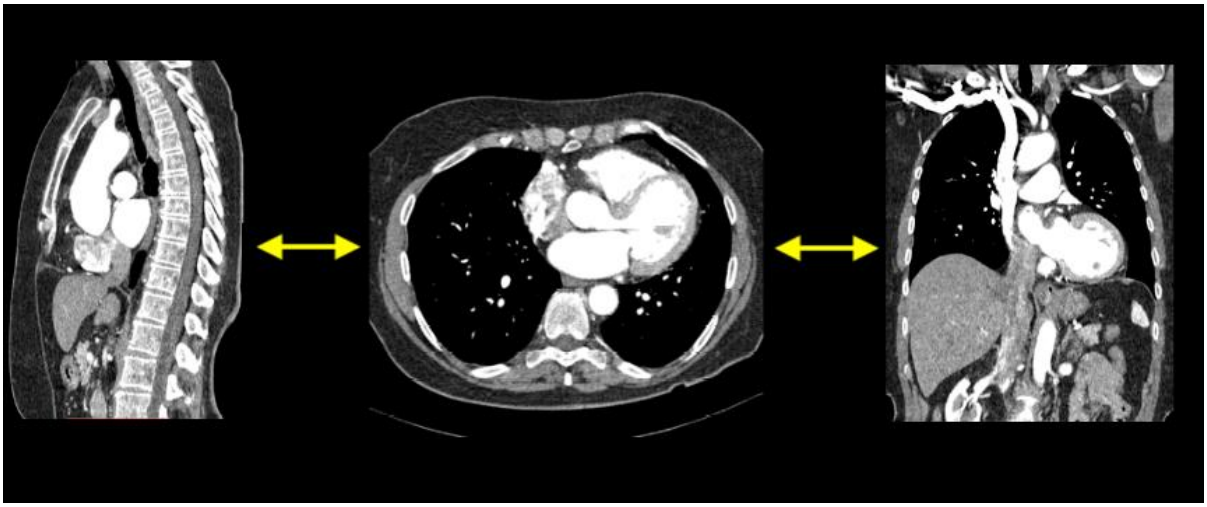
Στην πιο συνηθισμένη λοιπόν περίπτωση, το MPR περιλαμβάνει τη δημιουργία προοπτικών υπό κατάλληλες γωνίες από μια στοίβα Axial φετών, ώστε να μπορούν να παραχθούν εικόνες σε Coronal και Sagittal απεικονίσεις [36]. Στην Εικόνα 4-2 φαίνονται σχηματικά:



Εικόνα 4-2: Τα τρία κυριότερα επίπεδα του MPR

Αξίζει να αναφερθεί ότι υπάρχει και το Oblique view (λοξό), που δεν αναπαρίσταται από κάποιο συγκεκριμένο επίπεδο και απλώς έχει κάποια προσαρμοσμένη κλίση ως προς κάποιον άξονα, το οποίο στα πλαίσια αυτής της διπλωματικής (καθώς και στην πλειοψηφία των DICOM viewer) δεν εξετάστηκε.

Με αυτόν τον τρόπο, οι ειδικοί μπορούν να αξιοποιήσουν καλύτερα τα αποτελέσματα π.χ. ενός μαγνητικού τομογράφου και να επιτευχθεί απεικόνιση περισσότερης πληροφορίας από αυτή που αρχικά είχαν μεν, αλλά δε μπορούσαν να αξιοποιήσουν με τη συμβατική επεξεργασία δε. Στην Εικόνα 4-3 φαίνεται πως από τον αρχικό Axial προσανατολισμό μπορεί να εξαχθεί ο Coronal και ο Sagittal προσανατολισμός και ουσιαστικά να τριπλασιαστεί η παρεχόμενη πληροφορία στον ειδικό που χρησιμοποιεί τον DICOM viewer.



Εικόνα 4-3: Από τον Axial (μεσαίο) προσανατολισμό εξήχθη ο Sagittal (αριστερός) και ο Coronal (δεξής) προσανατολισμός

4.2.2 Η βιβλιοθήκη που βασιστήκαμε

Επειδή ο χώρος του Medical Imaging δραστηριοποιείται κυρίως σε standalone εφαρμογές για desktops και workspaces, επιλογές προηγμένης επεξεργασίας (όπως το MPR) για περιβάλλον browser σε open-source (ανοιχτού κώδικα) κοινότητες δεν υπάρχουν πολλές.

Εντούτοις, στα πλαίσια της διπλωματικής χρησιμοποιήθηκε η AMI.js [37]. Πρόκειται για μια εργαλειοθήκη Medical Imaging γραμμένη σε JavaScript από τον N. Rannou και μια ομάδα προγραμματιστών και contributors με τη βοήθεια του Boston Children's Hospital και του Harvard University. Διαθέτει τμηματική αρχιτεκτονική, ευέλικτο σχεδιασμό και φιλικό API, που σημαίνει ότι οι προγραμματιστές είναι ευπρόσδεκτοι στο να την παραμετροποιήσουν όπως επιθυμούν και να την εντάξουν στα προγράμματά τους, έχοντας φυσικά και άδεια εμπορικής χρήσης MIT. Αξιοποιεί στο έπακρο επιτάχυνση γραφικών από GPU, το οποίο πετυχαίνει μέσω της επίστρωσης του WebGL πάνω στην

προαναφερθείσα βιβλιοθήκη Three.js [38]. Αυτό, άλλωστε, ήταν και το κυρίαρχο κριτήριο επιλογής της AMI.js – η χρήση δηλαδή της Three.js – ώστε να επιτευχθεί μετέπειτα η συμβατότητα με τη SPA HERMES.

4.2.3 Υλοποίηση του MPR Demo

Όπως αναφέρθηκε, η βιβλιοθήκη AMI.js παρέχει την ευχέρεια στον προγραμματιστή να την παραμετροποιήσει κατάλληλα για την υλοποίηση δικών του προγραμμάτων. Έτσι, έγινε αντικατάσταση του custom loader (φορτωτή) και parser (μετατροπέα) της βιβλιοθήκης με αυτόν του HERMES, ο οποίος επιφέρει και καλύτερη απόδοση μιας και πραγματοποιείται σε περιβάλλον Web Worker.

Παράλληλα, ο DICOM Web Worker της πλατφόρμας HERMES ενισχύθηκε με έναν αλγόριθμο απόφασης για τη δυνατότητα πολυεπίπεδης επεξεργασίας του αρχείου εισόδου, καθώς και αντιστοίχισης στο αρχικό επίπεδο (σε περίπτωση που η πολυεπίπεδη επεξεργασία είναι εφικτή) με βάση τις τιμές των συνημίτονων κατεύθυνσης (direction cosines) των διανυσμάτων γραμμής και στήλης (row and columns vectors, v_r and v_c) του αρχείου εισόδου. Ολόκληρος ο κώδικας αυτού παρουσιάζεται παρακάτω:

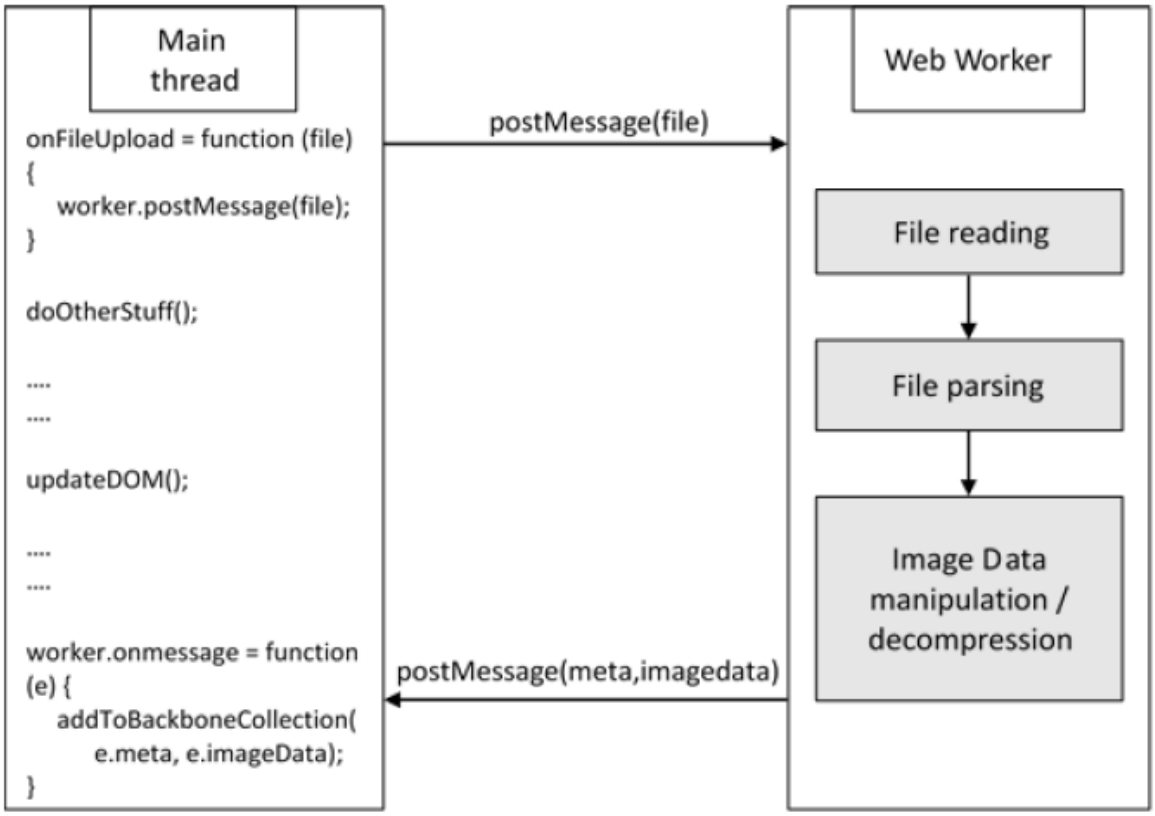
```
import DicomConverter from '../dicomTools/DicomConverter';

onmessage = function (evt) {
    let fileList = evt.data;
    let reader = new FileReaderSync();
    let file_index = 0;
    for (let i = 0; i < fileList.length; i++) {
        let dicomConverter = new DicomConverter();
        dicomConverter.convert(reader.readAsArrayBuffer(fileList[i]),
dicomObj => {
            dicomObj.associatedFile = fileList[i].name + '|' +
fileList[i].lastModified;
                  dicomObj.mpr = true; //flag for mpr possibility
            let imageOrientation, length;
            if  (dicomObj.header.x00200037 !== undefined) { //file series
                imageOrientation = dicomObj.header.x00200037.value;
                length = dicomObj.header.x00200037.value.length;
            }
            else { //(one) multi frame file
                if (dicomObj.header.x52009229 !== undefined) {
                    imageOrientation =
dicomObj.header.x52009229.value[0].x00209116.value[0].x00200037.value;
                    length =
dicomObj.header.x52009229.value[0].x00209116.value[0].x00200037.value.length
                }
            }
```

Εικόνα 4-4: Ο ανανεωμένος DICOM Web Worker (α μέρος)

```
                if (imageOrientation === undefined) {
                //mpr and volume rendering can't be applied
                    dicomObj.currentOrientation=0;
                    dicomObj.mpr = false;
                }
                else {
                //continue and find initial Plane
                    for (let k = 0 ; k < length; k = k + 1) {
                        imageOrientation[k] = Number(imageOrientation[k]);
                    }
                    const obliquityThresholdCosineValue = 0.8;
                    var axis = [null, null]; //axis[0] is for rowAxis, axis[1] is
for colAxis
                    var orientationX, orientationY ,orientationZ, absX, absY,
absZ;
                    for (let k = 0; k <= 1; k = k + 1) {
                        orientationX = imageOrientation[k*3] < 0 ? "R" : "L";
                        orientationY = imageOrientation[k*3+1] < 0 ? "A" : "P";
                        orientationZ = imageOrientation[k*3+2] < 0 ? "F" : "H";

                        absX = Math.abs(imageOrientation[k*3]);
                        absY = Math.abs(imageOrientation[k*3+1]);
                        absZ = Math.abs(imageOrientation[k*3+2]);

                        if (absX>obliquityThresholdCosineValue && absX>absY &&
absX>absZ) {
                            axis[k]=orientationX;
                        }
                        else if (absY>obliquityThresholdCosineValue && absY>absX
&& absY>absZ) {
                            axis[k]=orientationY;
                        }
                        else if (absZ>obliquityThresholdCosineValue && absZ>absX
&& absZ>absY) {
                            axis[k]=orientationZ;
                        }
                    }

                    if (axis[0] != null && axis[1] != null) {
                        if      (((axis[0] === "R") || (axis[0] === "L")) &&
((axis[1] === "A") || (axis[1] === "P"))) dicomObj.currentOrientation = 0;
                        else if (((axis[1] === "R") || (axis[1] === "L")) &&
((axis[0] === "A") || (axis[0] === "P"))) dicomObj.currentOrientation = 0;
                    //axial case

                        else if (((axis[0] === "R") || (axis[0] === "L")) &&
((axis[1] === "H") || (axis[1] === "F"))) dicomObj.currentOrientation = 2;
                        else if (((axis[1] === "R") || (axis[1] === "L")) &&
((axis[0] === "H") || (axis[0] === "F"))) dicomObj.currentOrientation = 2;
                    //coronal case

                        else if (((axis[0] === "A") || (axis[0] === "P")) &&
((axis[1] === "H") || (axis[1] === "F"))) dicomObj.currentOrientation = 1;
                        else if (((axis[1] === "A") || (axis[1] === "P")) &&
((axis[0] === "H") || (axis[0] === "F"))) dicomObj.currentOrientation = 1;
                    //sagittal case
                    }
```

Εικόνα 4-5: Ο ανανεωμένος DICOM Web Worker (β μέρος)

Με τη χρήση των worker, κάθε φορά που ο χρήστης επιλέγει να ανοίξει κάποιο αρχείο από κάποιον τοπικό φάκελο, το κύριο νήμα που είναι υπεύθυνο για την αλληλεπίδραση με το UI αποκτά πρόσβαση σε ένα File object που αντιστοιχεί στο αρχείο αυτό και κατόπιν το αποστέλλει μέσω της postMessage() κλήσης στο περιβάλλον ενός worker που έχει εκκινήσει με αποκλειστικό σκοπό την εκτέλεση του παραπάνω workflow. Σε περίπτωση ανοίγματος πολλαπλών αρχείων, πραγματοποιείται αντίστοιχος αριθμός κλήσεων/αποστολών, τα workflow των οποίων σειριοποιούνται, αλλά με μηδενικές επιπτώσεις στην αντιλαμβανόμενη εμπειρία χρήσης. Όταν η επεξεργασία λάβει τέλος, ο worker περνά την εξαχθείσα πληροφορία με τον ίδιο τρόπο στο κύριο νήμα, το οποίο είναι ελεύθερο να τη διαχειριστεί (Εικόνα 4-6).

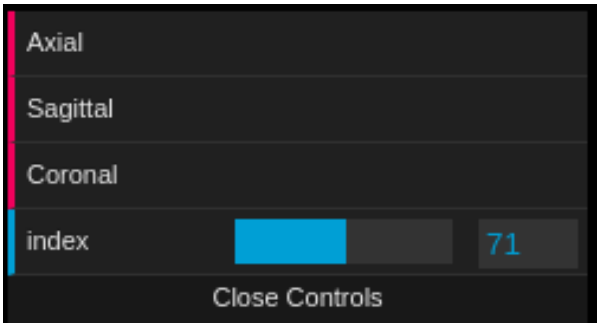


Εικόνα 4-6: Αλληλεπίδραση μεταξύ κύριου νήματος εκτέλεσης και Web Worker

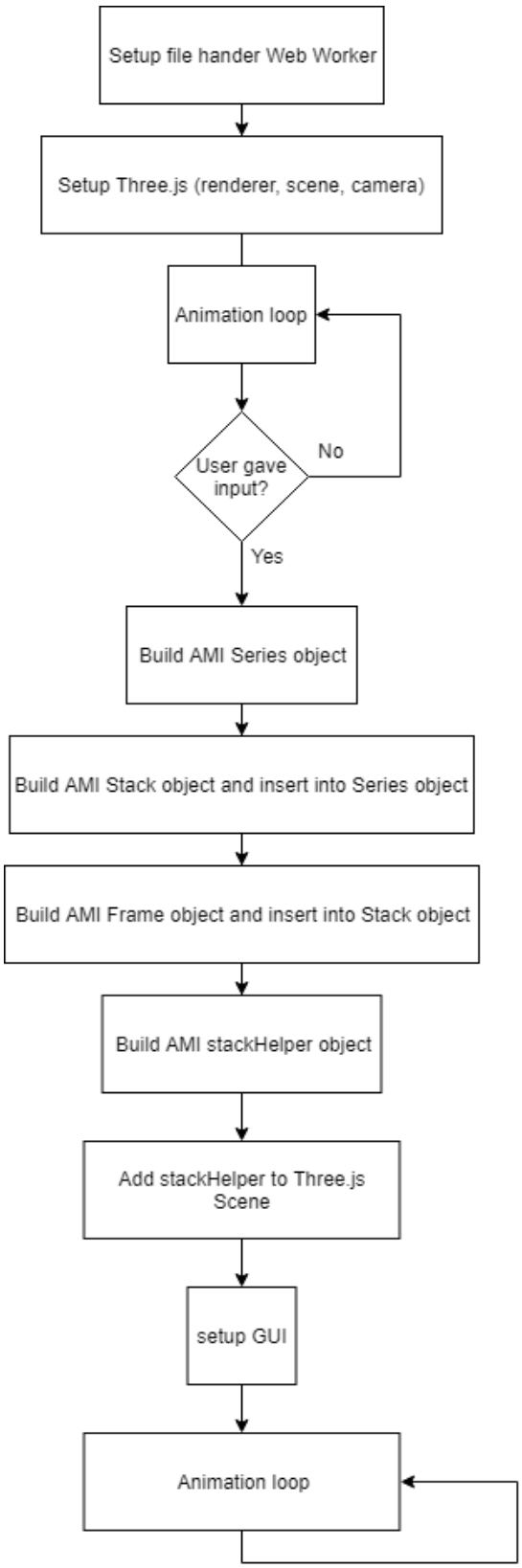
Για τη λειτουργία του προγράμματος, αρχικά δηλώνεται ο file handler ώστε να αναμένει είσοδο από το χρήστη, καθώς και ετοιμάζονται τα διάφορα API του Three.js. Έπειτα, το πρόγραμμα αναμένει την είσοδο του χρήστη, τρέχοντας παράλληλα το λεγόμενο animation loop (βρόχος επανάληψης που αναλαμβάνει το render κάθε προγράμματος Three.js). Όταν δοθεί η είσοδος εικόνων από το χρήστη, χτίζονται οι

διάφορες δομές του AMI και το γραφικό περιβάλλον της εφαρμογής. Τέλος, το πρόγραμμα τρέχει επαναληπτικά το animation loop ώστε να αναπαρίσταται στον χρήστη η ζητούμενη απεικόνιση. Σχηματικά δίνεται στην Εικόνα 4-8.

Παράλληλα, το τελικό πρόγραμμα δίνει στον χρήστη ένα μενού με τέσσερα κουμπιά, τρία για την επιλογή του επιπέδου και ένα slider για τη μετακίνηση μεταξύ των εικόνων. Αυτό υλοποιήθηκε με τη χρήση την open-source βιβλιοθήκης dat.GUI [39], η οποία είναι μια πολύ ελαφριά (lightweight) βιβλιοθήκη γραμμένη σε JavaScript (βλ. Εικόνα 4-7).



Εικόνα 4-7: Το μενού της demo εφαρμογής, που δίνει τη δυνατότητα στο χρήστη να επιλέξει προσανατολισμό και δείκτη σε εικόνα του συνόλου

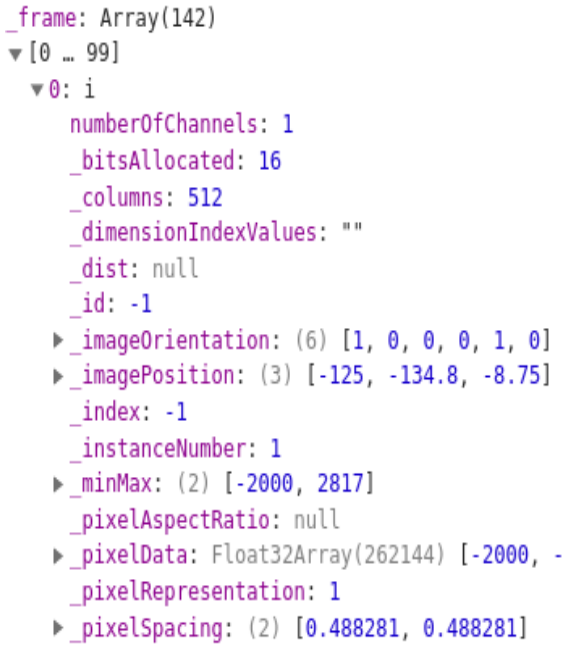


Εικόνα 4-8: Ροή του demo προγράμματος MPR

4.2.3.1 Οι δομές του AMI

Ακολουθεί μια σύντομη αναφορά των δομών του AMI, με τη σειρά εμφώλευσής τους:

- **Frame object:** Περιλαμβάνει πληροφορίες για την εικόνα του DICOM αρχείου, όπως θέση (position), προσανατολισμός (orientation), πληροφορίες pixel (pixel data) κλπ. Κάθε τέτοιο object αντιστοιχίζεται σε κάθε εικόνα του file series



Εικόνα 4-9: Frame object

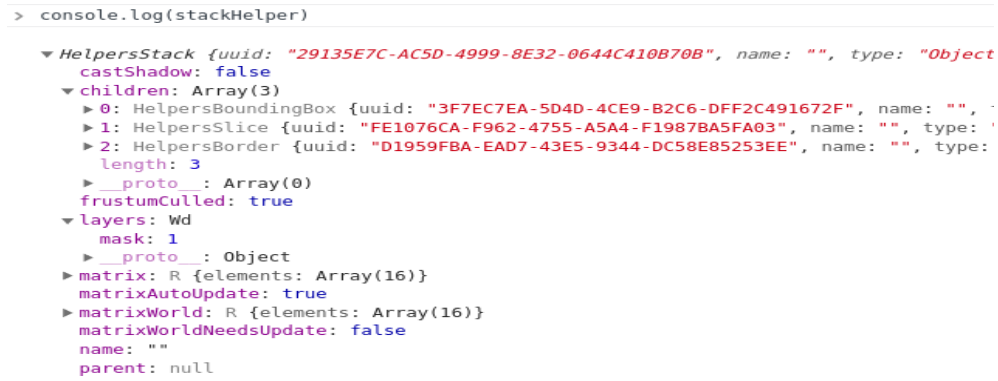
- **Stack object:** Περιλαμβάνει πληροφορίες για το σύνολο των εικόνων του file series, όπως διαστάσεις, μέγιστη/ελάχιστη θέση και φυσικά το σύνολο των Frame Objects

Εικόνα 4-10: Stack object

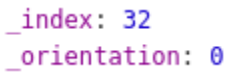
- **Series object:** Περιλαμβάνει πληροφορίες για τη συγκεκριμένη εξέταση, όπως όνομα ασθενή, ημερομηνία εξέτασης, κωδικό εξέτασης και φυσικά το Stack Object

Εικόνα 4-11: Series Object

- **StackHelper object:** Το object που τελικά προστίθεται στην απεικόνιση του Three.js. Είναι αντικείμενου τύπου Object3D, απαραίτητης μορφής που απαιτεί η βιβλιοθήκη αυτή και περιλαμβάνει, μεταξύ άλλων, το Stack object και τα πεδία index (δείκτης σε ποια εικόνα του συνόλου βρισκόμαστε) και orientation (δείκτης σε ποιο προσανατολισμό βρισκόμαστε).



Εικόνα 4-12: StackHelper object



Εικόνα 4-13: Τα δύο σημαντικά πεδία για την υλοποίηση

4.2.4 Το τελικό MPR Demo

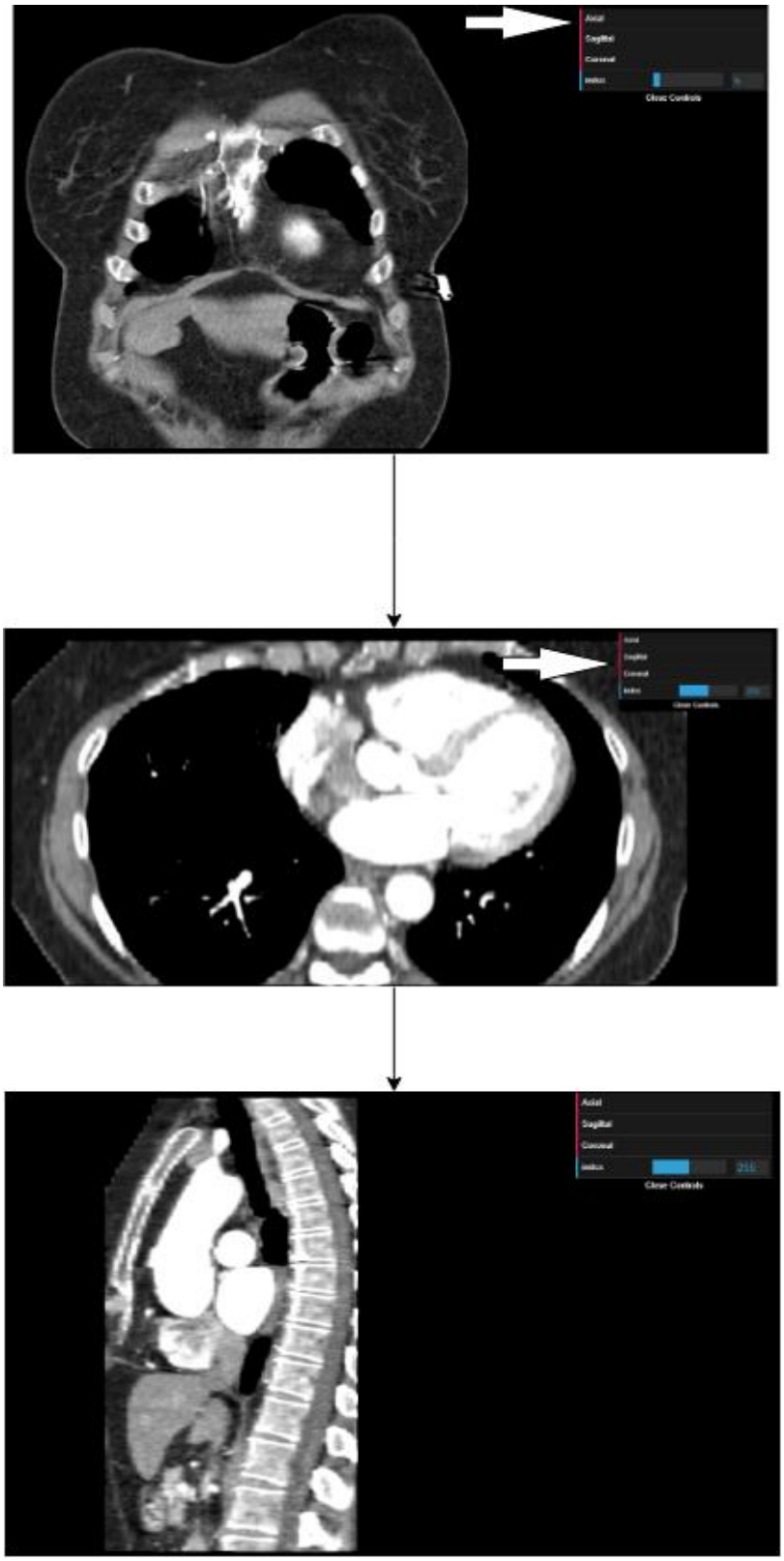
Η εφαρμογή αναπτύχθηκε σε περιβάλλον Node.js (Ενότητα 3.1.10) χρησιμοποιώντας τον Webpack development server για developing και testing, και αργότερα εξήχθη σε production με χρήστη του Webpack [40] ως bundler και του Babel [41] ως transpiller ώστε να εξαχθεί σε κώδικα JavaScript ES5 και να υποστηρίζεται και σε παλιότερες εκδόσεις browser.

Το τελικό αποτέλεσμα είναι ένας μαύρος container με ένα HTML input κουμπί. Όταν ο χρήστης πατήσει το κουμπί, καλείται να επιλέξει από τον τοπικό κατάλογο του υπολογιστή του ένα ή περισσότερα αρχεία τύπου DICOM (κατάληξη .dcm). Όταν το κάνει αυτό, το πρόγραμμα απεικονίζει τη μεσαία (στο σύνολό τους) εικόνα από τον αρχικό (default) προσανατολισμό, καθώς και το μενού του dat.GUI πάνω δεξιά.

Σε ένα ενδεικτικό παράδειγμα, δώσαμε ένα σύνολο 56 DICOM εικόνων με αρχικό προσανατολισμό Coronal. Επομένως, απεικονίστηκε αρχικά η 28η εικόνα του συνόλου. Με

τον slider για το index μετακινηθήκαμε στην 5η εικόνα. Έπειτα, επιλέξαμε να αλλάξουμε προσανατολισμό και να επιλέξουμε Αξονικό πατώντας το κουμπί Axial. Τέλος, επιλέξαμε και τον εναπομείναντα προσανατολισμό, Sagittal.

Εικόνα 4-14: Παράδειγμα χρήσης του MPR Demo (α μέρος): μετακίνηση του slider

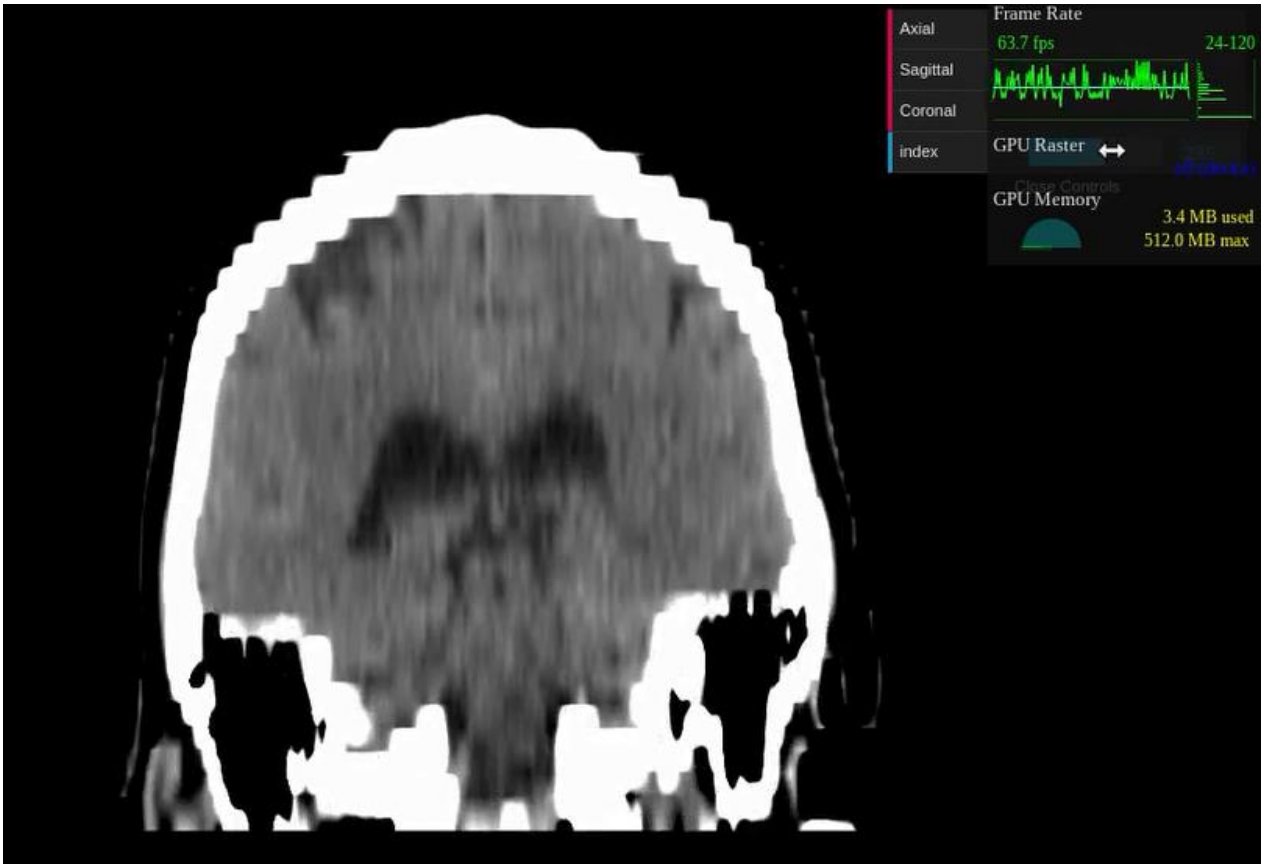


Εικόνα 4-15: Παράδειγμα χρήσης του MPR Demo (β μέρος): Επιλογή διαφορετικού προσανατολισμού

Εύκολα επίσης μπορεί να παρατηρήσει κανείς πως αλλάζουν τα άκρα των τιμών που μπορεί να πάρει το index. Δηλαδή, αφού αρχικά είχαμε 56 εικόνες (και index) διαστάσεων 512 x 512, στα άλλα δύο επίπεδα θα έχουμε 512 εικόνες (άρα και index).

4.2.5 Επιδόσεις

Σε μια εφαρμογή, ιδιαίτερα σε μια δικτυακή, εφάμιλλης σημασίας της αποτελεσματικότητας μιας εφαρμογής είναι και η απόδοσή της, ανεξάρτητα αν αυτή «τρέχει» σε ισχυρούς ή όχι υπολογιστές. Η συγκεκριμένη λοιπόν υλοποίηση, σε δοκιμή με πληθώρα διαφορετικών υπολογιστών απέφερε σταθερά επιδόσεις καρέ 60fps (60 καρέ ανά δευτερόλεπτο) και άνω (όπως μετρήθηκε από εργαλείο που παρέχει ο Google Chrome προς τους προγραμματιστές [42]), καθιστώντας την πολύ αποδοτική. Βέβαια, η υπολογιστική εργασία της πολυεπίπεδης ανασύνθεσης δεν αποτελεί εξίσου βαρύ φόρτο εργασίας για ένα τερματικό υπολογιστή όπως π.χ. η απεικόνιση τρισδιάστατων μοντέλων (βλ. Ενότητα 4.3).



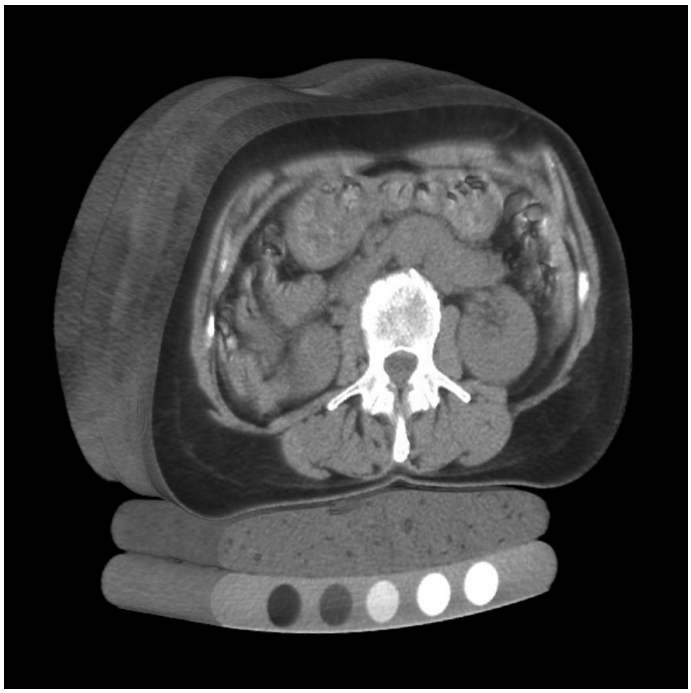
Εικόνα 4-16: Στιγμιαίο αποτέλεσμα των fps όπως φαίνεται από το αντίστοιχο εργαλείο του Chrome Dev Tools

4.3 Απεικόνιση Τρισδιάστατων Μοντέλων – 3D Volume Rendering

4.3.1 Το Volume Rendering στην πράξη

Στην επιστημονική απεικόνιση και στο σχεδιασμό γραφικών του υπολογιστή, το Volume Rendering είναι ένα σύνολο τεχνικών που χρησιμοποιούνται για την απεικόνιση μιας δισδιάστατης προβολής ενός τρισδιάστατου διακριτού συνόλου δειγμάτων, συνήθως ενός τρισδιάστατου βαθμωτού πεδίου.

Ένα τυπικό τρισδιάστατο σύνολο δεδομένων είναι μια ομάδα δισδιάστατων εικόνων (φέτες) που έχουν αποκτηθεί από έναν υπολογιστικό τομογράφο ή ένα μαγνητικό τομογράφο. Συνήθως αυτά αποκτώνται σε κάποιο κανονικό μοτίβο (π.χ. μια φέτα κάθε χιλιοστό) και έχουν έναν φυσιολογικό αριθμό εικονοστοιχείων εικόνας σε κανονικό μοτίβο. Στην Εικόνα 4-17 φαίνεται ένα παράδειγμα ενός κανονικού ογκομετρικού πλέγματος, με κάθε στοιχείο όγκου (voxel, από το volume και το pixel) που αντιπροσωπεύεται από μία μόνο τιμή που λαμβάνεται δειγματοληπτικά από την άμεση περιοχή που περιβάλλει το voxel.

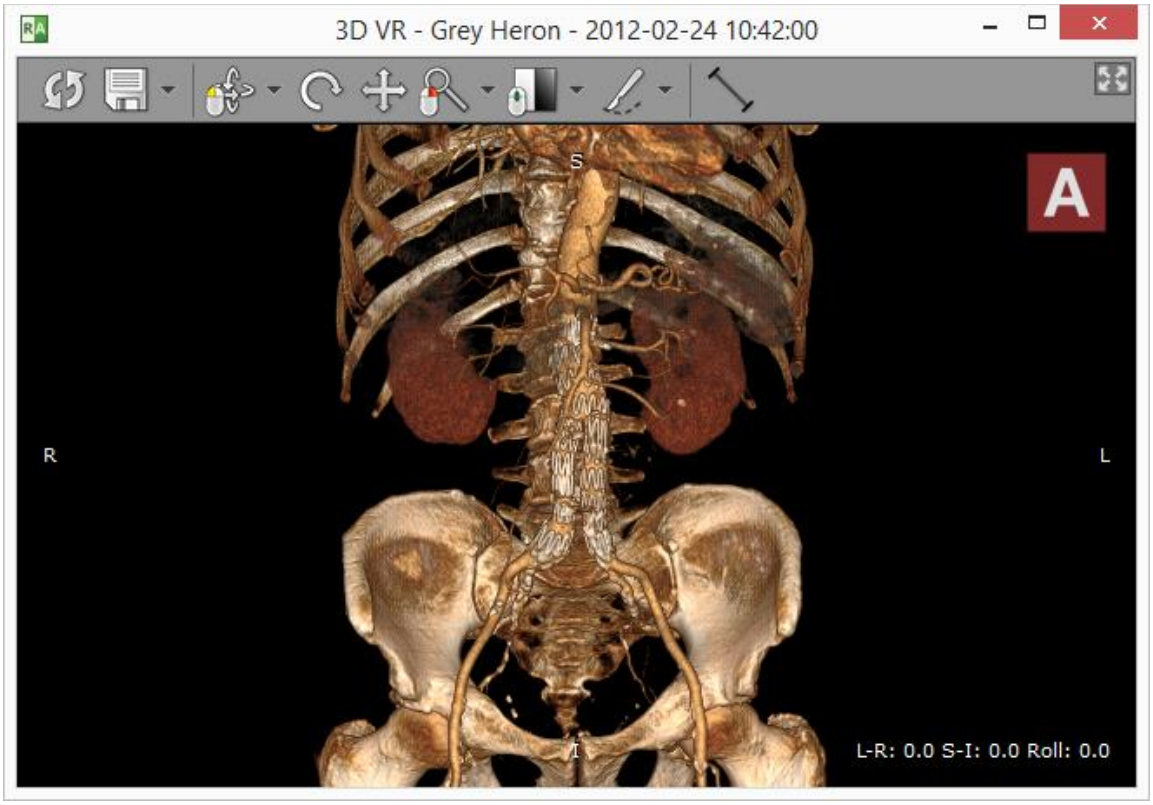


Εικόνα 4-17: Ογκομετρικό πλέγμα που εξήχθη από X-ray τομογράφο

Για να απεικονιστεί μια 2D προβολή του τρισδιάστατου συνόλου δεδομένων, πρέπει πρώτα να οριστεί μια κάμερα στο χώρο σε σχέση με τον όγκο. Επίσης, πρέπει να οριστεί η αδιαφάνεια και το χρώμα κάθε voxel. Αυτό συνήθως ορίζεται χρησιμοποιώντας μια RGBA (για το κόκκινο, πράσινο, μπλε, διαφάνεια - alpha) συνάρτηση μεταφοράς που καθορίζει την RGBA τιμή για κάθε πιθανή τιμή voxel.

Για παράδειγμα, ένας όγκος μπορεί να προβληθεί με την εξαγωγή ισοεπιφανειών (επιφάνειες ίσων τιμών) από τον όγκο και την απεικόνισή τους ως πολυγωνικά πλέγματα ή με την απεικόνιση του όγκου απευθείας ως ένα μπλοκ δεδομένων. Ο αλγόριθμος των macrhing cubes είναι μια κοινή τεχνική για την εξαγωγή μιας ισοεπιφάνειας από τα δεδομένα όγκου. Η άμεση απόδοση του όγκου είναι μια εργασία που αποτελεί βαρύ υπολογιστικό φόρτο και μπορεί να εκτελεστεί με διάφορους τρόπους [43].

Για την απεικόνιση όγκων από Medical Imaging προγράμματα πρέπει και εδώ τα αρχεία εισόδου να τηρούν κάποια κριτήρια, όπως το να αποτελούν μια συλλογή από DICOM αρχεία της ίδιας εξέτασης που θα έχουν το ίδιο πάχος και ύψος μεταξύ τους και θα είναι παράλληλα μεταξύ τους.



Εικόνα 4-18: Volume Rendering μιας σειράς DICOM, όπως φαίνεται από το πρόγραμμα Radiant [44]

4.3.2 Η βιβλιοθήκη που βασιστήκαμε

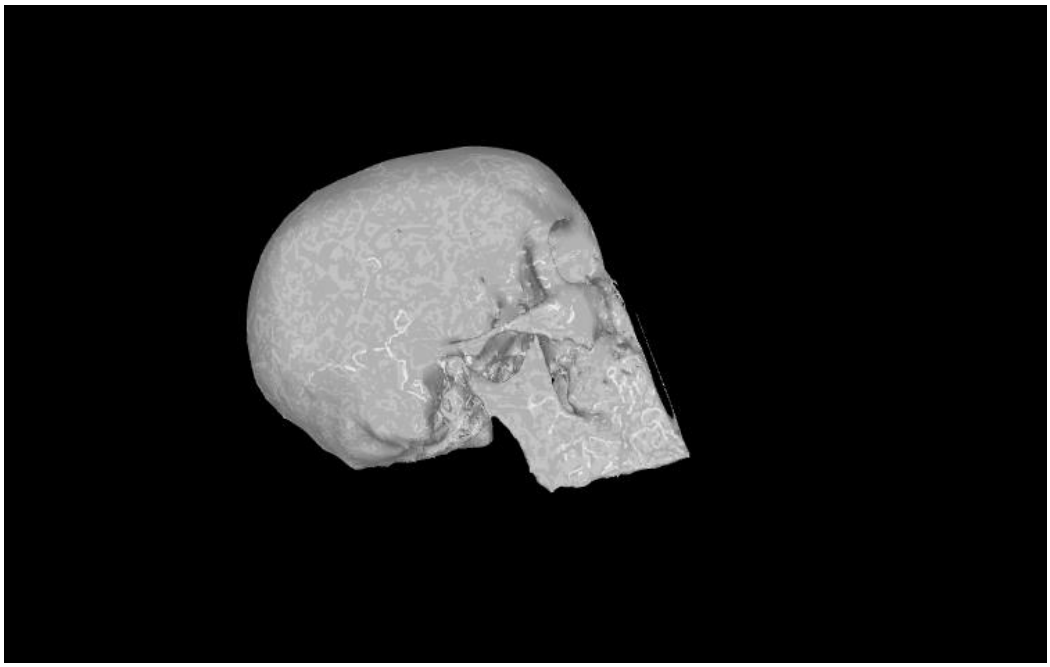
Σε αντίθεση με την προηγούμενη υλοποίηση, στο Volume Rendering οι open-source βιβλιοθήκες στο διαδίκτυο που επιτελούν τη ζητούμενη λειτουργία είναι περισσότερες. Παρ' όλ' αυτά, η τελική επιλογή έπρεπε να γίνει με τα εξής κριτήρια (και με σειρά προτεραιότητας):

- **Απόδοση:** ως μια τόσο βαριά υπολογιστική εργασία και για ένα πρόγραμμα που απευθύνεται σε γιατρούς για χρήση μέσω οποιασδήποτε συσκευής (αρχή BYOD – bring your own device) στην κατοχή τους, το πρόγραμμα πρέπει να ανταποκρίνεται πλήρως, στο μέγιστο δυνατό βαθμό, στις πλέον παρωχημένες συσκευές
- **Αποτέλεσμα:** επίσης σημαντικό είναι το εξαγόμενο αποτέλεσμα να αποτελεί την πλέον ξεκάθαρη απεικόνιση του δείγματος εισόδου, χωρίς να το παραμορφώνει ή/και να προσθέτει επιπλέον λεπτομέρειες. Χρήσιμο (ως και ζητούμενο) επίσης πρόσθετο είναι η παροχή στο χρήστη της δυνατότητας να ρυθμίζει 1) την αδιαφάνεια των voxel και 2) την ποιότητα του εξαγόμενου όγκου, ανάλογα με τις επιδόσεις της συσκευής του
- **Παραμετροποίηση:** η παρεχόμενη δυνατότητα στον προγραμματιστή ώστε αυτός να μπορεί να διαχειριστεί τη βιβλιοθήκη και να την προσαρμόσει στις ανάγκες και κατευθύνσεις που ο ίδιος κρίνει ότι χρειάζονται (π.χ. επιλογή χρωματικής παλέτας)

Παρακάτω παρουσιάζονται βιβλιοθήκες που εξετάστηκαν.

4.3.2.1 X Toolkit

Το XTK (ή X Toolkit) [45] είναι μια ισχυρή βιβλιοθήκη ανοιχτού κώδικα που υλοποιεί τρισδιάστατες απεικονίσεις από διάφορους τύπους αρχείων, όχι μόνο DICOM. Αν και ικανοποιητικά αποδοτική, η μειωμένη δυνατότητα παραμετροποίησης της εξόδου (σε σχέση με άλλες βιβλιοθήκες) και κατά συνέπεια του ζητούμενου αποτελέσματος δεν επέτρεψαν την επιλογή της.



Εικόνα 4-19: Αποτέλεσμα τρισδιάστατης απεικόνισης DICOM από το XTK

4.3.2.2 AMI.js Singlepass example

Αποτελεί λειτουργία της βιβλιοθήκης AMI που χρησιμοποιήθηκε για την πολυεπίπεδη κατασκευή. Παρόλο που ήταν εύκολη η ενσωμάτωσή της διότι ακολουθούσε την ίδια προγραμματιστική λογική με την προηγούμενη υλοποίηση (βασισμένη σε Three.js) και μπορούσε να παραμετροποιηθεί, εντούτοις οι επιδόσεις της ακόμη και για μικρά δείγματα DICOM ήταν απογοητευτική, πιάνοντας περί τα 10 καρέ το δευτερόλεπτο (fps) κατά μέσο όρο.

4.3.2.3 VTK.js

Την καλύτερη λοιπόν επιλογή αποτέλεσε η VTK.js [46]. Η βιβλιοθήκη αυτή προσφέρει ένα σύνολο από χρήσιμες δυνατότητες για το rendering δεδομένων στο διαδίκτυο όντας γρήγορη και εύρωστη. Διατηρεί τις ίδιες έννοιες και αρχιτεκτονική με τη δημοφιλή VTK C++ βιβλιοθήκη και έχει συνταχθεί από την Kitware [47], μία μεγάλη εταιρία στο χώρο του Medical Imaging και του Medical Computing γενικότερα.

Η βιβλιοθήκη αυτή παρέχει εύχρηστο και πλήρως παραμετροποιήσιμο API, εξαιρετικές επιδόσεις, επιλογές χρωματικής παλέτας και δυνατότητες παροχής στο χρήστη επιλογής της αδιαφάνειας των voxel και της ποιότητας της εξόδου. Έτσι, ο κάθε ειδικός που

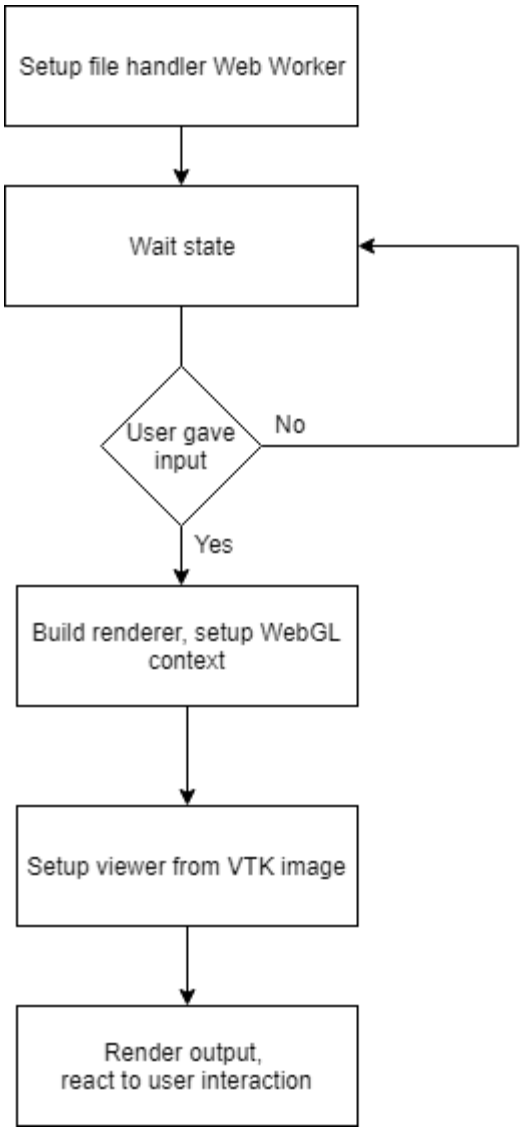
χρησιμοποιεί τον DICOM viewer μπορεί να προσαρμόζει δυναμικά την ποιότητα εξόδου του προγράμματος στις επιδόσεις του μηχανήματός του. Τέλος, η βιβλιοθήκη είναι γραμμένη σε pure JavaScript και δεν έχει εξαρτήσεις από άλλες μεγάλες βιβλιοθήκες ή frameworks.

4.3.3 Υλοποίηση του Volume Rendering Demo

Ως βάση της υλοποίησης αποτέλεσε η βιβλιοθήκη vtk.js για το rendering καθώς και η δομή του εξαγόμενου αντικειμένου της βιβλιοθήκης itk.js, χωρίς ωστόσο να γίνει χρήση της τελευταίας. Για την εξαγωγή του επιθυμητού αποτελέσματος έγιναν κάποιες σχεδιαστικές επιλογές με σημαντικότερες τις παρακάτω:

- Η απεξάρτηση από την itk.js με χρήση του loader + parser της SPA HERMES. Παρ' όλ' αυτά, διατηρήθηκε η δομή του object που εξάγει η itk, αλλά αυτό δημιουργήθηκε χωρίς τη χρήση της
- Η προσαρμογή του PiecewiseGaussianWidget (που είναι υπεύθυνο για το χτίσιμο του μενού του GUI της εφαρμοφής) ώστε 1) να αφαιρεθούν τα πολλά και περιττά controls, 2) να διατηρηθούν αυτά για διαφάνεια και ποιότητα εξόδου, 3) να γίνει αλλαγή της καμπύλης που αναπαριστά τα βάρη στα δεδομένα της εισόδου με ένα ορθογώνιο, ώστε να απεικονίζεται πιο ομοιόμορφα το αποτέλεσμα και 4) να αφαιρεθεί η περιττή παροχή στον χρήστη να το προσαρμόσει ή να προσθέσει/αφαιρέσει επιπλέον, μειώνοντας έτσι την πολυπλοκότητα χρήσης του
- Η αλλαγή της προεπιλεγμένης χρωματικής παλέτας με μια με πιο ουδέτερες αποχρώσεις

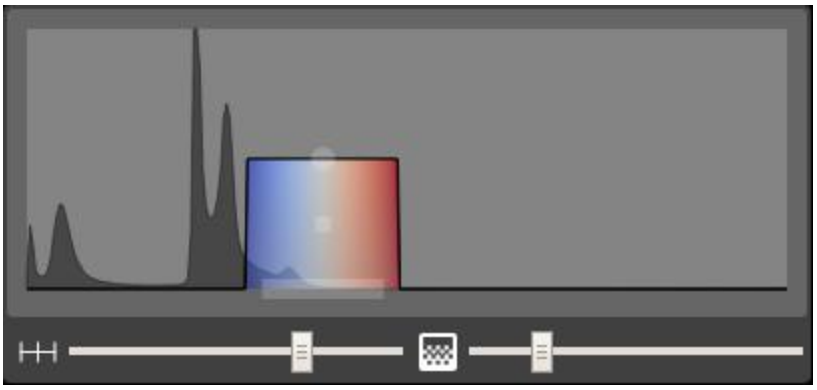
Αρχικά, δηλώνεται ο file handler ώστε να αναμένει είσοδο από το χρήστη. Όταν δοθεί η είσοδος εικόνων, χτίζεται η δομή του itk object με τη αξιοποίηση κατάλληλων πεδίων των DICOM αρχείων, το οποίο στη συνέχεια μετατρέπεται σε vtk object με χρήση βοηθητικής συνάρτησης της vtk.js. Έπειτα, χτίζεται ο renderer του vtk και η διεπαφή χρήστη, δίνοντας ως παραμέτρους διάφορες σχεδιαστικές επιλογές που έγιναν. Στο στάδιο αυτό επίσης η vtk αναλαμβάνει τη δημιουργία του WebGL context και την αλληλεπίδραση με αυτό. Τέλος, το πρόγραμμα τρέχει συνεχώς το rendering loop και φροντίζει παράλληλα να «αντιδρά» σε πιθανή αλληλεπίδραση του χρήστη με το πρόγραμμα. Σχηματικά δίνεται στην Εικόνα 4-20.



Εικόνα 4-20: Ροή του demo προγράμματος Volume Rendering

Το τελικό αποτέλεσμα του μενού του γραφικού περιβάλλοντος (GUI) φαίνεται στην εικόνα 4-21, όπου ο χρήστης μπορεί:

a) να μετακινεί το ορθογώνιο κατά μήκος του οριζοντίου άξονα (slider) και τα τονίζει διαφορετικές χρωματικές λεπτομέρειες του αντικειμένου, ξεκινώντας από αριστερά από τις μαύρες αποχρώσεις και πηγαίνοντας στα δεξιά προς τις λευκές αποχρώσεις
b) να ρυθμίζει την πυκνότητα των voxel και συνεπώς την ποιότητα της εξόδου (κάτω αριστερά) και
c) να ρυθμίζει την αδιαφάνεια του όγκου (κάτω δεξιά)



Εικόνα 4-21: Τα controls του GUI menu

Παράλληλα ο χρήστης μπορεί με τη χρήση του ποντικιού πάνω στο αντικείμενο να κάνει τις εξής ενέργειες:

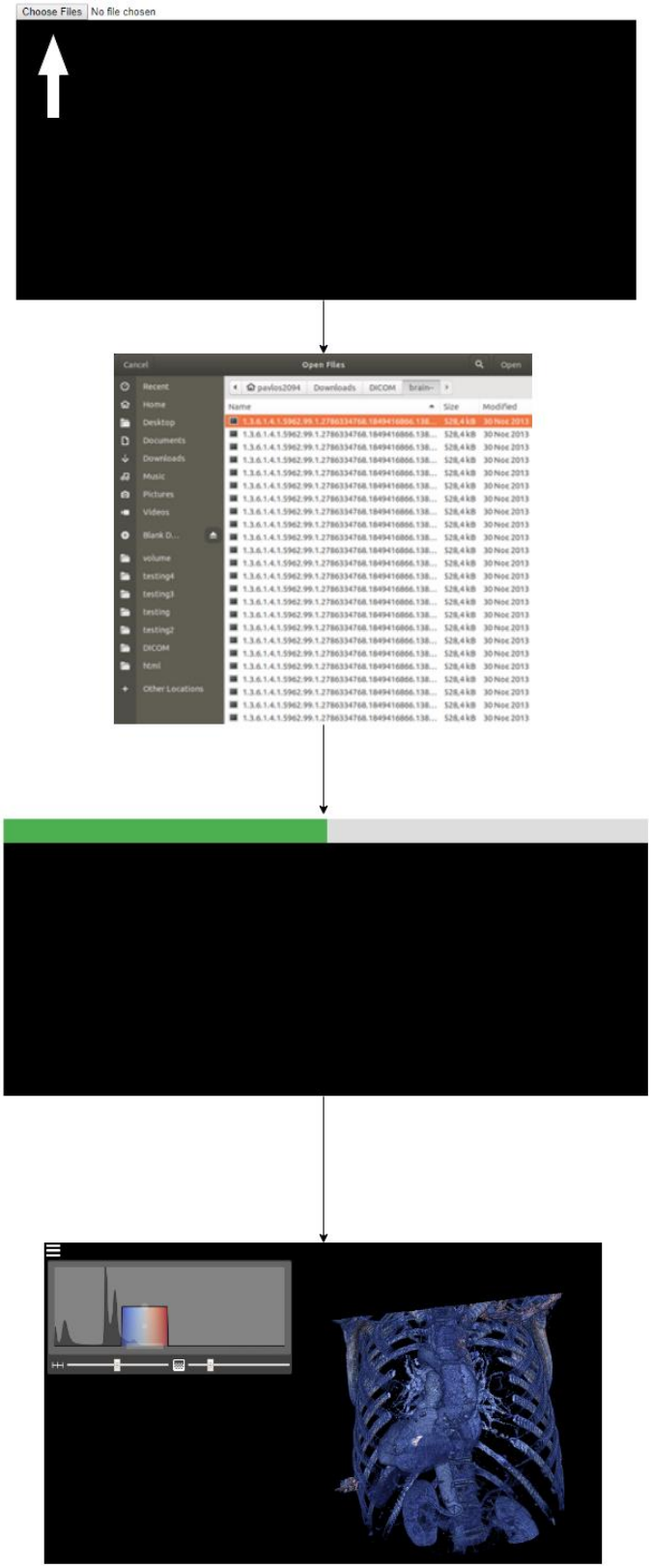
a) Να το περιστρέφει (rotate) με αριστερό κλικ
b) Να το μετακινεί (pan) με δεξί κλικ
c) Να κάνει μεγέθυνση προς τα μέσα ή προς τα έξω (zoom in/zoom out) με αριστερό κλικ έχοντας πατημένο το κουμπί Ctrl του πληκτρολογίου

4.3.4 Το τελικό Volume Rendering Demo

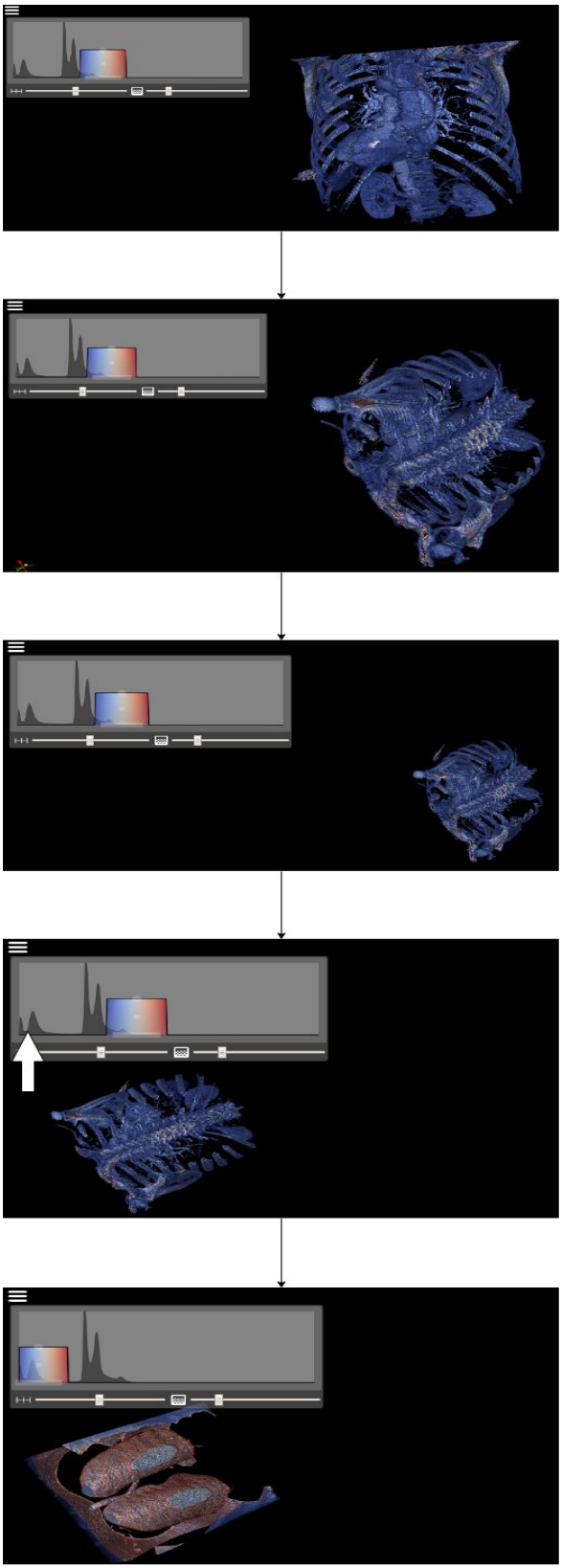
Η εφαρμογή αναπτύχθηκε σε περιβάλλον Node.js (Ενότητα 3.1.10) χρησιμοποιώντας τον Webpack development server για developing και testing, και αργότερα εξήχθη σε production με χρήστη του Webpack [40] ως bundler και του Babel [41] ως transpiller ώστε να εξαχθεί σε κώδικα JavaScript ES5 και να υποστηρίζεται και σε παλιότερες εκδόσεις browser.

Το τελικό αποτέλεσμα είναι ένας μαύρος container με ένα HTML input κουμπί. Όταν ο χρήστης πατήσει το κουμπί, καλείται να επιλέξει από τον τοπικό κατάλογο του υπολογιστή του ένα ή περισσότερα αρχεία τύπου DICOM (κατάληξη .dcm). Όταν το κάνει αυτό το πρόγραμμα απεικονίζει μια μπάρα φόρτωσης των αρχείων, η οποία όταν ολοκληρωθεί απεικονίζει την ογκομετρική απεικόνιση της σειράς αρχείων DICOM μαζί με το μενού του GUI.

Σε ένα ενδεικτικό παράδειγμα, δώσαμε το ίδιο σύνολο 56 DICOM εικόνων. Στη συνέχεια μετά το loading, δοκιμάσαμε την περιστροφή, το σμίκρυνση τη μετακίνηση του αντικειμένου και τη μετακίνηση του slider για τονισμό διαφορετικών λεπτομερειών.



Εικόνα 4-22: Παράδειγμα χρήσης του Volume Rendering Demo (α μέρος)

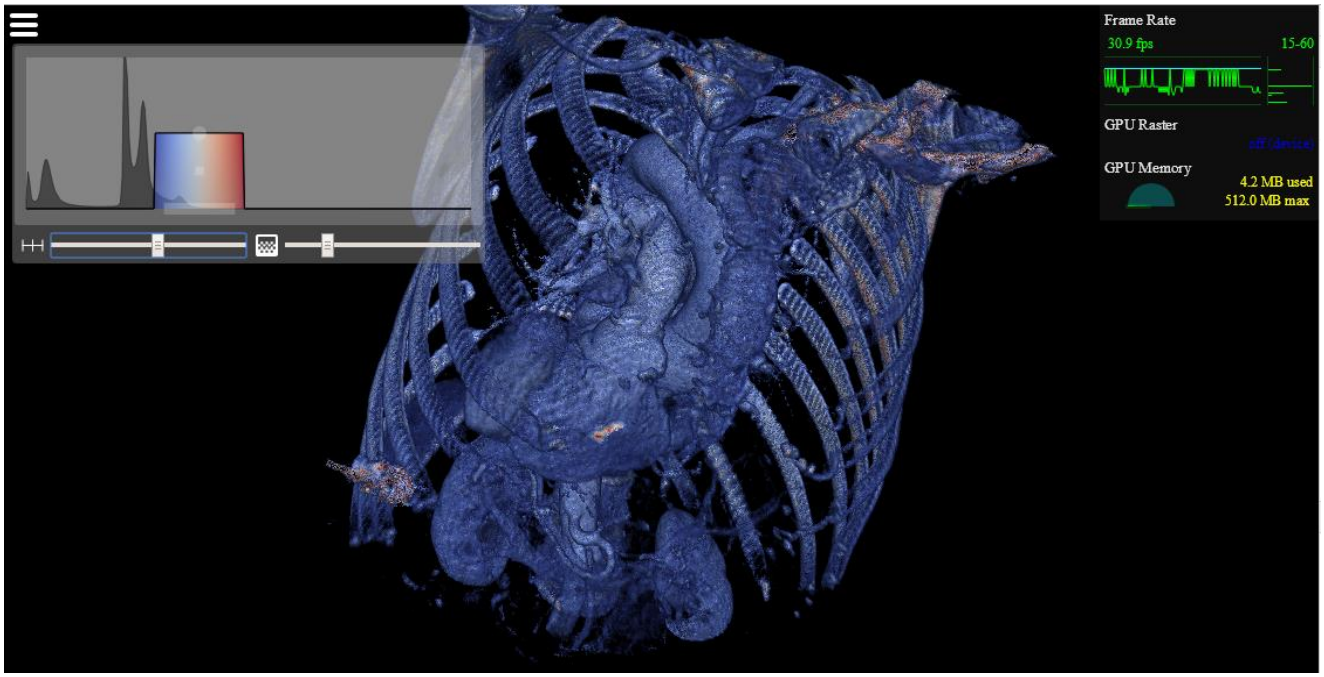


Εικόνα 4-23: Παράδειγμα χρήσης του Volume Rendering Demo (β μέρος): περιστροφή, σμίκρυνση, μετακίνηση του αντικειμένου, μετακίνηση του slider

4.3.5 Επιδόσεις

Εξαρχής το θέμα στην υλοποίηση του Volume Rendering Demo ήταν η επίτευξη ικανοποιητικών επιδόσεων, λαμβάνοντας υπ' όψη το βαρύ υπολογιστικό φόρτο της ογκομετρικής απεικόνισης. Εντούτοις, με τη παρεχόμενη δυνατότητα στο χρήστη να προσαρμόσει τα γραφικά ώστε να βρει την κατάλληλη ισορροπία με την απόδοση, το εμπόδιο αυτό προσπερνάται επιτυχώς.

Ενδεικτικά, σε έναν φορητό υπολογιστή μέτριων επιδόσεων (επεξεργαστής Intel I5 2^{ης} γενιάς, κάρτα γραφικών NVIDIA GeForce GT 540M) με τον δείκτη της πυκνότητας των pixel να είναι κάπου στη μέση σε ένα δείγμα 56 εικόνων DICOM παρουσιάστηκε επίδοση περί τα 30 καρέ το δευτερόλεπτο (fps) στη χρήση και εναλλαγή μεταξύ των λειτουργιών της εφαρμογής.



Εικόνα 4-24: Στιγμιαίο αποτέλεσμα των fps όπως φαίνεται από το αντίστοιχο εργαλείο του Chrome Dev Tools

5 Ενσωμάτωση στη SPA HERMES

5.1 Σκοπός του κεφαλαίου

Με την ολοκλήρωση των ανεξάρτητων demo που υλοποιούν τις λειτουργίες για πολυεπίπεδη ανασύνθεση και τρισδιάστατη απεικόνιση, επόμενο βήμα για την περάτωση της εργασίας ήταν το Integration με την Single Page Application HERMES. Ενός ολοκληρωμένου επαγγελματικού εργαλείου στα κορυφαία πρότυπα, που σαφώς αποτέλεσε μεγαλύτερη πρόκληση. Στα επόμενα κεφάλαια θα γίνει ανάλυση των κοινών σχεδιασμών και υλοποιήσεων που έπρεπε να γίνουν, καθώς ξεχωριστή αναφορά μετέπειτα για κάθε προστιθέμενη λειτουργικότητα.

5.2 Σύντομη αναφορά στη δομή του προγράμματος

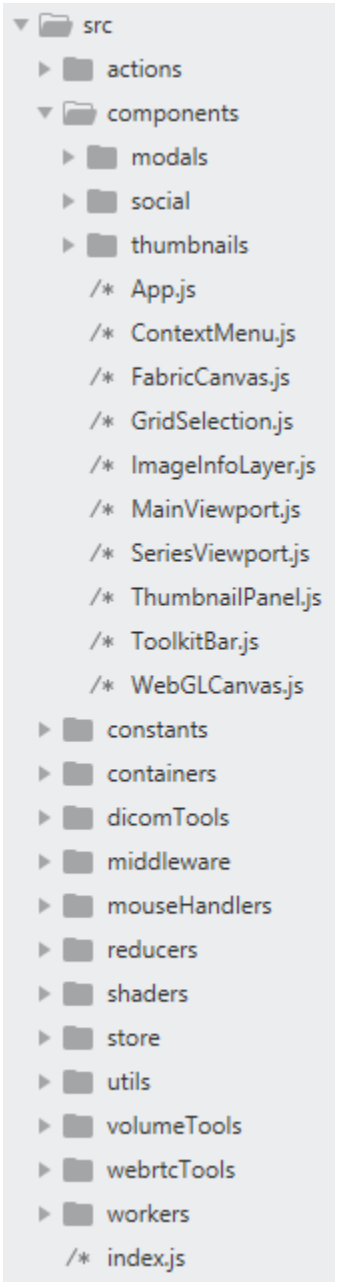
Όπως έχει ήδη αναφερθεί, η SPA HERMES είναι γραμμένη σε React μαζί με Redux. Σε ό,τι αφορά το πρώτο, αυτό σημαίνει ότι το πρόγραμμα είναι χωρισμένο σε διακριτά Component τα οποία αναλαμβάνουν το χτίσιμο του UI, καλώντας για το σκοπό αυτό και άλλα Component. Επομένως, έπρεπε σε κάθε στάδιο του development να τροποποιηθούν τα υπάρχοντα ή να δημιουργηθούν καινούργια ώστε να γίνουν οι απαραίτητες τροποποιήσεις, κρατώντας παράλληλα την ήδη υπάρχουσα λειτουργικότητα της εφαρμογής.

Σε ότι αφορά το δεύτερο, η χρήση του Redux αυτόματα σημαίνει ότι η κατάσταση της εφαρμογής “ζει” στο Store. Δηλαδή, οποιαδήποτε αλληλεπίδραση του χρήστη με την εφαρμογή σημάνει κάποιο Action (δράση), το οποίο στη συνέχεια φτάνει στους Reducers που αναλαμβάνουν να τροποποιήσουν το Store, άρα και το State. Προφανώς και η δομή αυτή είναι που ευνοεί την πτυχή της συνεργατικότητας της εφαρμογής, το να μπορούν δηλαδή τα διάφορα Actions να γίνονται dispatch στους συνομιλητές αξιοποιώντας παράλληλα διαύλους WebRTC. Επομένως, απαραίτητη ήταν επίσης η δημιουργία νέων Actions καθώς και Reducers που θα αναλαμβάνουν το χειρισμό αυτών.

Σχετικά με την πολυεπίπεδη ανασύνθεση, η προσθήκη αυτής της λειτουργικότητάς επηρεάζει σε σημαντικό βαθμό την υπάρχουσα απεικόνιση που αναλαμβάνεται από το Canvas του WebGL, καθώς το κύριο “παράθυρο” της απεικόνισης θα πρέπει να αντικατασταθεί από το καινούργιο, το οποίο θα προσφέρει και επιπλέον δυνατότητες. Αντίθετα, η τρισδιάστατη απεικόνιση σημαίνει ουσιαστικά καμία κατάργηση υπαρχόντων μονάδων και απλώς προσθήκη Components και Modal (παράθυρο), μιας και αντίστοιχη κοντινή δυνατότητα δεν υπήρχε προηγουμένως.

5.3 Ενσωμάτωση της Πολυεπίπεδης Ανασύνθεσης

Η κύρια δομή του καταλόγου φακέλων του προγράμματος είναι η παρακάτω:



Εικόνα 5-1: Δομή του καταλόγου φακέλου του προγράμματος

Το Component το οποίο αναλαμβάνει την απεικόνιση του Canvas στοιχείου είναι το WebGLCanvas.js, στο οποίο και έγιναν οι περισσότερες τροποποιήσεις. Με τη χρήση της βιβλιοθήκης AMI.js, αντικαταστάθηκαν δομές όπως η Camera του Three.js με μια επαυξημένων δυνατοτήτων, ενώ πλέον το αντικείμενο που προστίθεται στο Scene του Three.js είναι το stackHelper object που παράγει η βιβλιοθήκη, μέσα στο οποίο έχουν δημιουργηθεί οι απεικονίσεις και των άλλων δύο συμπληρωματικών επιπέδων. Η εναλλαγή μεταξύ εικόνων και επιπέδων γίνεται με τα πεδία index και orientation του stackHelper, αντίστοιχα.

Παράλληλα, τροποποιήθηκαν διάφορες μέθοδοι του Component, όπως το πότε θα γίνεται το rendering με τα νέα ορίσματα (μιας και πλέον “ακούμε” και για αλλαγή επιπέδου), καθώς και ο χειριστής αλλαγής μεγέθους παραθύρου (resize handler) και η πλειοψηφία των μεθόδων επεξεργασίας της ιατρικής εικόνας.

Σε αυτό το σημείο πρέπει να αναφερθεί ότι, πέρα από το κομμάτι της συνεργατικότητας, η client εφαρμογή λειτουργεί σαν ένας πλήρης λειτουργιών standalone DICOM viewer. Μερικές δυνατότητες που διαθέτει για το σκοπό αυτό είναι οι ακόλουθες:

- Μετακίνηση εικόνας
- Μεγέθυνση/σμίκρυνση εικόνας
- Αλλαγή προσανατολισμού
- Βασικά ιατρικά φίλτρα για επεξεργασία εικόνας (φωτεινότητα, αντίθεση, αρνητικό φίλτρο κτλ.)
- Εργαλεία μέτρησης σημείων ενδιαφέροντος (annotations) πάνω στην εικόνα (μήκους, επιφάνειας, γωνίας, κτλ.)

Γίνεται λοιπόν εύκολα αντιληπτό ότι η αλλαγή του κύριο παράθυρου απεικόνισης των ιατρικών εικόνων θα σήμαινε και κατάλληλη προσαρμογή των δυνατοτήτων αυτών για τη συνέχεια της υποστήριξης τους. Πράγματι, έγιναν οι εξής τροποποιήσεις στις αντίστοιχες μεθόδους:

- Γίνεται μετακίνηση της θέσης του camera object (και όχι του scene) πλέον, ενώ παράλληλα φροντίζουμε να μετακινούμε και το επίπεδο z, εφόσον ανήκει στο χώρο που βρισκόμαστε
- Έγινε διαχωρισμός για τη μεγέθυνση εικόνων μεταξύ των ποσοστιαίων επιλογών (100%, 200%, 400%) και του μεγεθυντικού φακού, χρησιμοποιώντας για την περίπτωση του πρώτου τη μέθοδο fitBox του camera του AMI
- Χρησιμοποιείται πλέον η μέθοδος rotate του camera του AMI για την περιστροφή της εικόνας, αντί για το πεδίο rotation του scene
- Γίνεται αλλαγή των window center και window width μέσω των αντίστοιχων πεδίων του SliceObject που ανήκει στο stackHelper
- Έγινε αλλαγή της αντιστοίχισης των annotations σε εικόνες. Έτσι, από εκεί που αντιστοιχίζονταν από τον κωδικό του ονόματός τους (SOP Instance UID) πλέον

αντιστοιχίζονται σε μια ακολουθία που περιλαμβάνει τον αριθμό της εικόνας (index) και του επιπέδου (orientation), ακολουθούμενο από τον κωδικό SOP Instance UID

Επιπλέον, έγιναν προσθήκες πεδίων στο αντικείμενο series (μεταξύ άλλων), ώστε να παρέχεται πληροφορία για τις διαστάσεις των επιπέδων που δεν είχαμε αρχικά άλλα δημιουργήθηκαν στο πλαίσιο του MPR. Ακόμη, προστέθηκαν τα κατάλληλα actions για την επιλογή αλλαγής επιπέδου (orientation), reducers που αναλαμβάνουν την αλλαγή αυτή και ένδειξη κατάστασης (σε ποιο επίπεδο βρισκόμαστε, εφόσον καθίσταται δυνατό) στην εργαλειοθήκη με τροποποιήσεις στα κατάλληλα Component.

5.4 Ενσωμάτωση της Απεικόνισης Τρισδιάστατων Μοντέλων

Για την ενσωμάτωση της τρισδιάστατης απεικόνισης στο HERMES, έπρεπε να δημιουργηθεί ένα Modal Component το οποίο παρέχει η ReactJS για την απεικόνιση πληροφοριών ή γραφικών σε αναδυόμενο παράθυρο. Το Modal καλείται από τον κεντρικό Container (Component που αναλαμβάνει και data fetching) κατά το πάτημα του αντίστοιχου κουμπιού από το χρήστη, το οποίο είναι επιλέξιμο εφόσον η σειρά DICOM αρχείων παρέχει τις απαραίτητες πληροφορίες για να μπορεί να πραγματοποιηθεί η τρισδιάστατη απεικόνιση.

Στη συνέχεια, το αναδυόμενο Modal αρχικά καλεί ένα παράθυρο με ένα loading spinner, το οποίο θα απεικονίζεται όσο γίνεται το rendering του Component που αμέσως μετά καλείται. Το ViewerCreator Component που καλείται χρησιμοποιεί το API του vtk.js και κάνοντας διάφορα calls σε αυτό αρχικά δημιουργεί το vtkImage που απαιτεί ο renderer του VTK. Τέλος, γίνεται build το UI της εφαρμογής μέσω του οποίου μπορεί να αλληλοεπιδρά ο χρήστης και απεικονίζεται το τρισδιάστατο μοντέλο.

Είναι πολύ σημαντικό να αναφερθεί ότι για να γίνει εφικτή η αφομοίωση του Volume Renderer με το WebRTC, απαραίτητη προϋπόθεση ήταν να γίνει αντιστοίχιση των event listeners για τις κινήσεις του ποντικιού (είτε στο κύριο παράθυρο του Modal είτε στο Widget) με Redux actions που δημιουργήθηκαν και Redux reducers, ώστε να μπορεί κατά την αλληλεπίδραση ενός χρήστη με αυτό να γίνεται dispatch και στον συνομιλητή του η ίδια δράση.

Γίνεται λοιπόν μια αναφορά των listeners που έχουν χρησιμοποιηθεί για την αλληλεπίδραση του χρήστη με το Modal. Αυτοί που αφορούν το Widget είναι οι εξής:

- Πάτημα κουμπιού ποντικιού (mousedown) στο widget, ενημερώνει το widgetHoverState

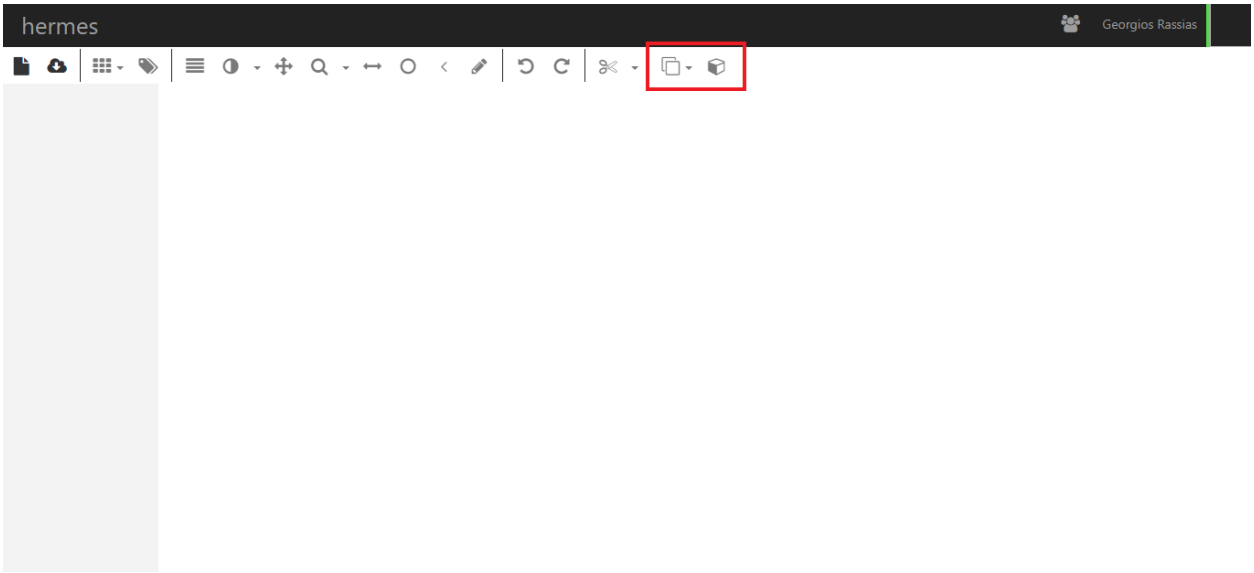
- Κίνηση ποντικιού (mousemove) επί του widget, αποστέλλεται μόνο εφόσον είναι πατημένο το κουμπί του ποντικιού, ενημερώνει το widgetBrowseState
- Άφημα κουμπιού ποντικιού (mouseup και mouseout), σηματοδοτεί τη λήξη του γεγονότος κίνησης της συνάρτησης μεταφοράς του Widget χωρίς ωστόσο να κάνει dispatch κάποιο action (μόνο τοπικά μέσα στο Component)
- Κύλιση της μπάρας pixel spacing, που επηρεάζει και την ποιότητα του εξαγόμενου μοντέλου. Εδώ κρίθηκε σκόπιμο να μη γίνεται dispatch το action μεταξύ των συνομιλητών, διότι ο κάθε χρήστης δύναται να χρησιμοποιεί διαφορετικών δυνατοτήτων μηχάνημα και επομένως θα θέλει να προσαρμόσει την έξοδο στις δυνατότητες του δικού του υπολογιστή
- Κύλιση της μπάρας αδιαφάνειας, ενημερώνει το widgetOpacityState

Ενώ αυτοί που αφορούν το κύριο παράθυρο του Modal:

- Πάτημα κουμπιού ποντικιού (mousedown), ενημερώνει το downstate
- Κίνηση του ποντικιού (mousemove) επί του κύριου παράθυρου, αποστέλλεται μόνο εφόσον είναι πατημένο το κουμπί του ποντικιού, ενημερώνει το moveState
- Άφημα κουμπιού ποντικιού (mouseup και mouseleave), σηματοδοτεί τη λήξη του αντίστοιχου γεγονότος (περιστροφή, μετακίνηση ή μεγέθυνση/σμίκρυνση, ενημερώνει το upState

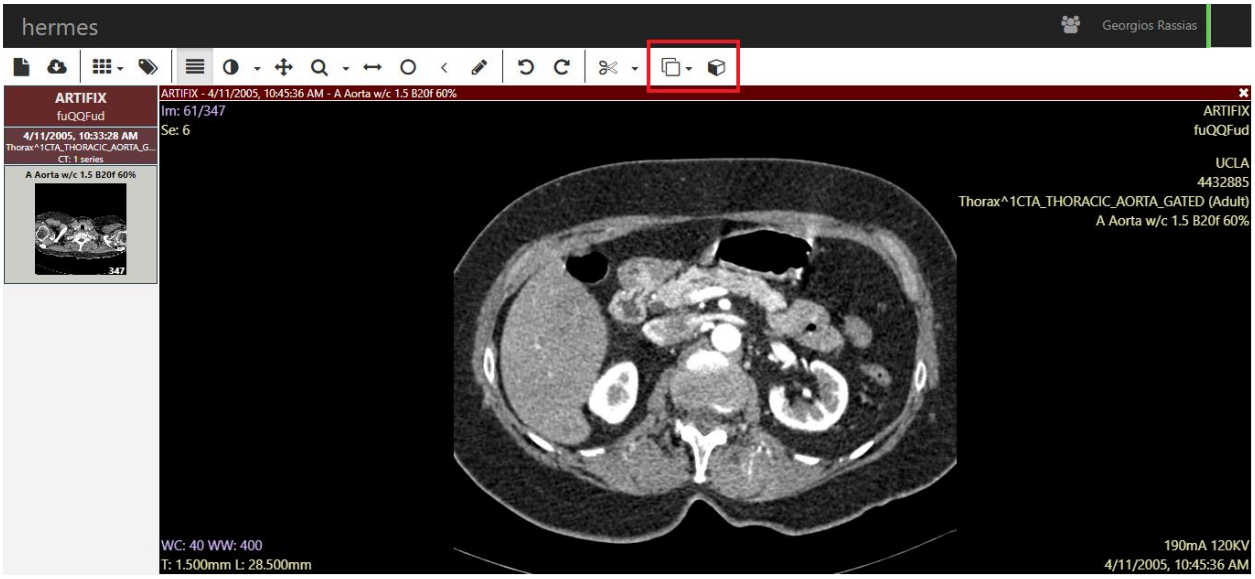
5.5 Το τελικό σύστημα

Το τελικό πρόγραμμα φαίνεται παρακάτω. Στην αρχική σελίδα της εφαρμογής πελάτη έχουν προστεθεί στα δεξιά της μπάρας εργαλείων δύο κουμπιά για τις λειτουργικότητες που προστέθηκαν αντίστοιχα.



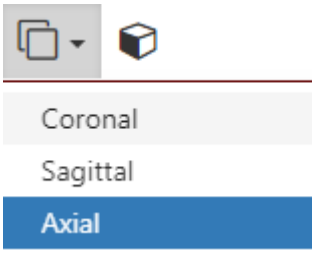
Εικόνα 5-2: Η αρχική ανανεωμένη διάταξη της SPA, με τα δύο νέα κουμπιά

Όταν ο χρήστης προσθέσει τα αρχεία του και εφόσον είναι εφικτή η επεξεργασία για MPR και Volume Rendering, τα κουμπιά αυτά γίνονται επιλέξιμα.



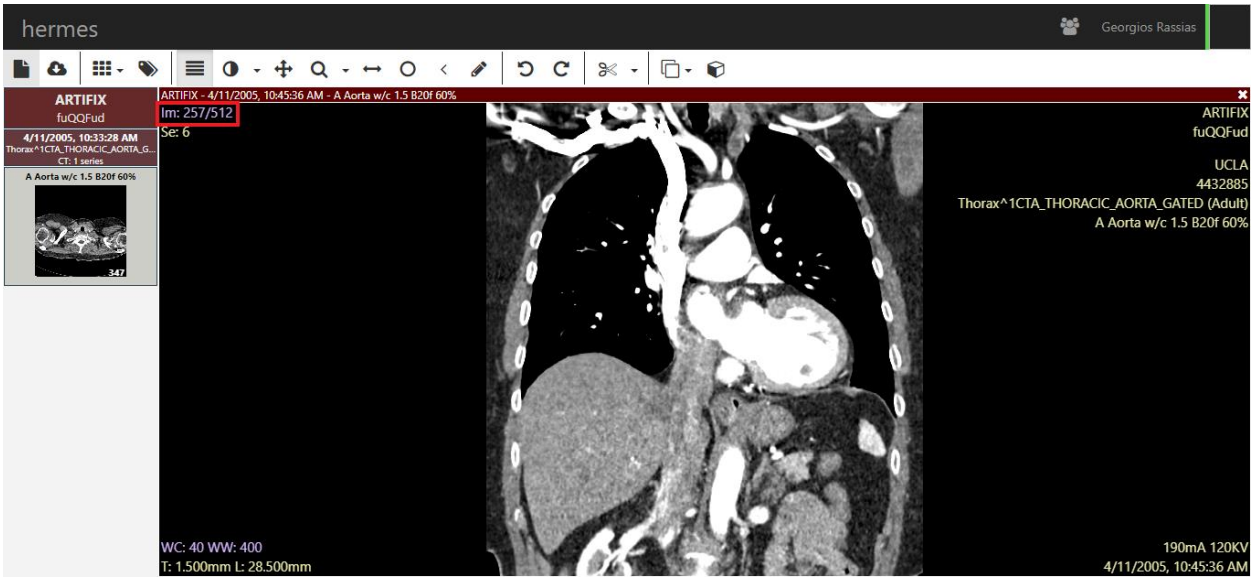
Εικόνα 5-3: Με την είσοδο αρχείων από το χρήστη, τα κουμπιά γίνονται επιλέξιμα

Το πάτημα του πλήκτρου για το MPR ανοίγει ένα πλαίσιο διαλόγου, στο οποίο ο χρήστης επιλέγει το επίπεδο που επιθυμεί να απεικονιστεί.



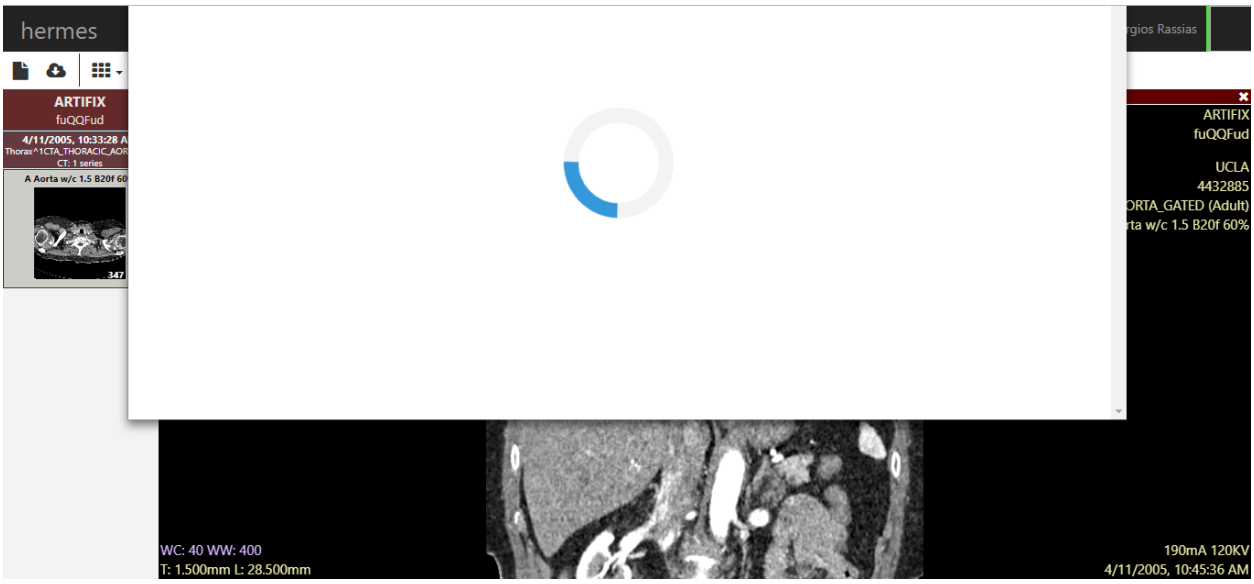
Εικόνα 5-4: Το πλαίσιο διαλόγου για την επιλογή επιπέδου

Με την επιλογή κάποιου άλλου επιπέδου, στον κύριο παράθυρο την εφαρμογής απεικονίζεται το επιλεγμένο επίπεδο. Μπορούμε να διακρίνουμε το πως μέγιστο δυνατό index έχει αλλάξει, κατ' αναλογία με τις διαστάσεις την αρχικής εικόνας.



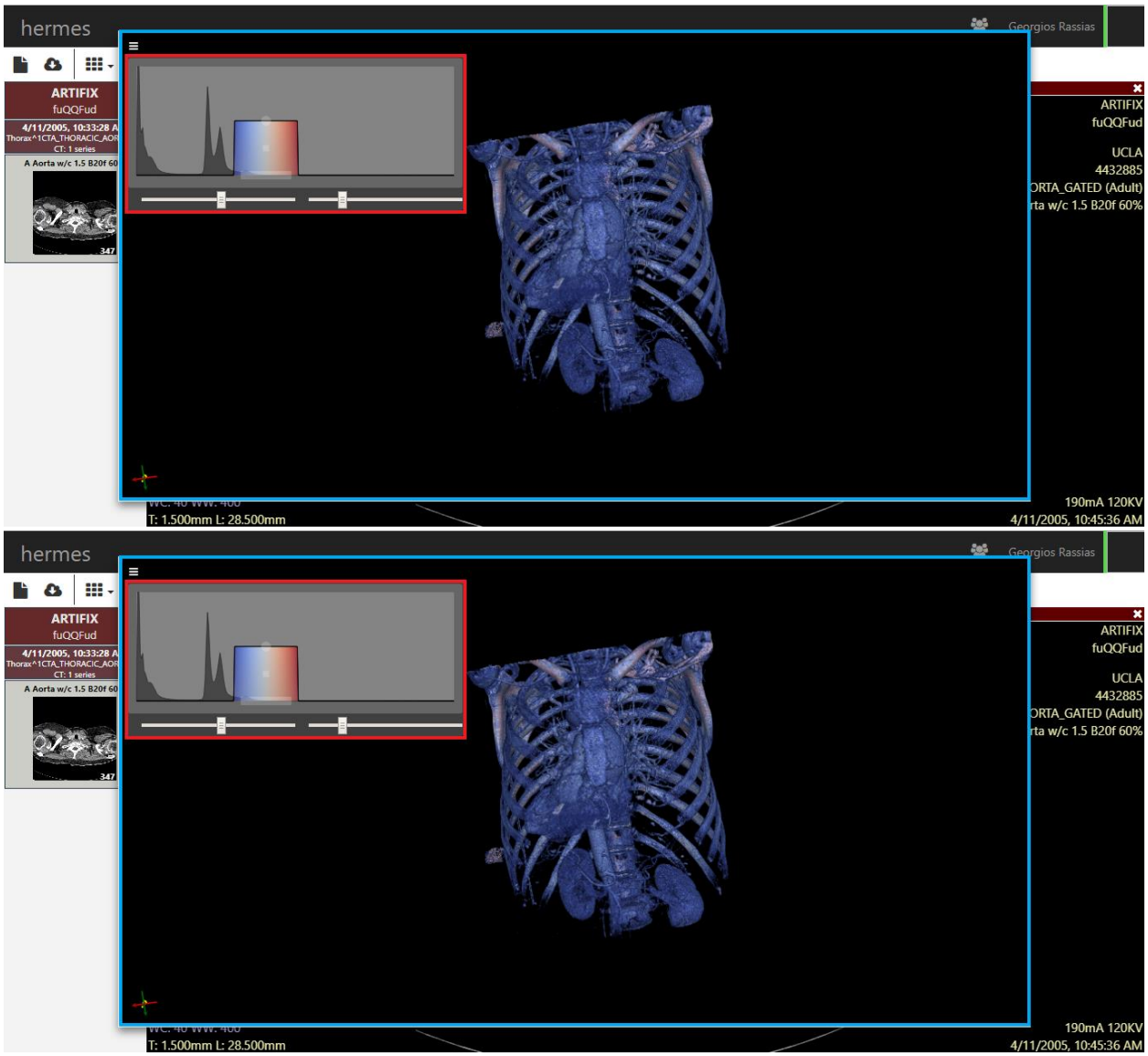
Εικόνα 5-5: Μετά την επιλογή νέου επιπέδου, αυτό απεικονίζεται ενώ διακρίνεται και το ανανεωμένο Index

Τέλος, με την επιλογή για την τρισδιάστατη απεικόνιση εμφανίζεται στην αρχή το loading spinner:



Εικόνα 5-6: Loading spinner κατά το φόρτωμα της τρισδιάστατης απεικόνισης

Ενώ μετά το φόρτωμα αυτού απεικονίζεται το κύριο παράθυρο του Volume Modal (με μπλε) και το Widget για τη συνάρτηση μεταφοράς (με κόκκινο):



Εικόνα 5-7: Το Volume Modal. Με μπλε φαίνεται το κύριο παράθυρο αυτού και με κόκκινο το Widget για τη συνάρτηση μεταφοράς

6 Επίλογος

6.1 Σύνοψη

Η παρούσα διπλωματική εργασία επιχείρησε να προσεγγίσει τις εξελικτικές ανάγκες τις συνεργατικής πλατφόρμας HERMES και να βελτιώσει την ποιότητα υπηρεσιών που αυτή προσφέρει στους χρήστες της. Τόσο η ενσωμάτωση της δυνατότητας για πολυεπίπεδη

ανασύνθεση αλλά κυρίως για απεικόνιση τρισδιάστατων μοντέλων, μιας εργασίας με βαρύ υπολογιστικό φόρτο, σε “ελαφρύ” δικτυακό περιβάλλον κρίνεται επιτυχημένη και όταν μάλιστα αυτή μπορεί να πραγματοποιείται σε πραγματικό χρόνο. Μπορεί λοιπόν απρόσκοπτα να συνεχιστεί η BYOD (Bring Your Own Device) λογική, έχοντας παράλληλα όλη την απαιτούμενη επεξεργασία να λαμβάνει χώρα στην πλευρά του πελάτη (client-side) και αποφεύγοντας έτσι την άσκοπη επικοινωνία πάνω από το δίκτυο τη στιγμή που αξιοποιείται στο μέγιστο η επεξεργαστική ικανότητα των σύγχρονων συσκευών. Το τελευταίο αυτό στοιχείο σε συνδυασμό με τη peer-to-peer δικτύωση που χρησιμοποιείται από την πλατφόρμα για τη διασύνδεση των χρηστών, καθιστά το εργαλείο αρκετά κλιμακώσιμο και αποδοτικό προκειμένου να υποστηρίξει απομακρυσμένα MDTMs.

6.2 Μελλοντικοί στόχοι

Η ενίσχυση της πλατφόρμας σε ότι αφορά το DICOM viewer κομμάτι της επετεύχθη, ωστόσο θα μπορούσε να γίνει διερεύνηση για δυνατότητες 3D MPR (παράλληλη απεικόνιση και των τριών επιπέδων σε ένα παράθυρο και ταυτόχρονη απεικόνιση του αντίστοιχου σημείου του χώρου σε κάθε επίπεδο) ή/και image fusion στο ίδιο πλαίσιο.

Παράλληλα, σε ό,τι αφορά το κομμάτι της συνεργατικότητας έχει γίνει αναφορά [48] ότι θα μπορούσε να γίνει ενσωμάτωση CAD (Computer-Aided Diagnosis) δυνατοτήτων με σκοπό την περαιτέρω υποστήριξη της διαγνωστικής διαδικασίας. Έχει δειχθεί πως η ανάλυση περιοχών ενδιαφέροντος (ROIs) και η κατάταξή τους σε σημασιολογικά οργανωμένες ταξινομίες με χρήση αλγορίθμων μηχανικής μάθησης μπορεί να αυξήσει τις πιθανότητες ανακάλυψης καινοτόμου γνώσης μέσω της δημιουργίας νέων υποθέσεων [49]. Με αυτόν τον τρόπο, οι πόροι που θα γίνονται διαθέσιμοι θα είναι σε θέση να διευκολύνουν ιατρικό προσωπικό και ερευνητές διαφορετικών ειδικεύσεων να εντοπίσουν συσχετισμούς μεταξύ του δικού τους πεδίου ενδιαφέροντος και εκείνων τα οποία έχουν ήδη μελετηθεί και ταξινομηθεί.

7 Bibliography

[1] T. Robertson, "The Public Availability of Actions and Artefacts," in *Computer Supported Cooperative Work (CSCW)*, 2002, pp. 299-316.

[2] [Online]. Available: www.hl7.org.

[3] "Bitmap - Wikipedia, the free encyclopedia," [Online]. Available: https://en.wikipedia.org/wiki/Bitmap.

[4] C. Roobottom, G. Mitchell and G. Morgan-Hughes, "Radiation-reduction strategies in cardiac computed tomographic angiography," in *Clinical Radiology*, 2010, pp. 859-867.

[5] R. Shreiber, 3-D Reconstruction in Radiology.

[6] "W3Schools," [Online]. Available: https://www.w3schools.com/css/css_intro.asp.

[7] "W3Schools," [Online]. Available: https://www.w3.org/Style/CSS20/history.html.

[8] "Auth0," [Online]. Available: https://auth0.com/blog/a-brief-history-of-javascript/.

[9] "Techopedia," [Online]. Available: https://www.techopedia.com/definition/23240/browser- compatibility.

[10] "WindowTimers.setTimeout() - Web API Interfaces," [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/API/WindowOrWorkerGlobalScope/setTimeout.

[11] "WindowTimers.setInterval() - Web API Interfaces | MDN," [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/API/WindowOrWorkerGlobalScope/setInterval.

[12] "XMLHttpRequest - Wikipedia, the free encyclopedia," [Online]. Available: https://en.wikipedia.org/wiki/XMLHttpRequest.

[13] "SharedWorker - Web API Interfaces | MDN," Mozilla Foundation, [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/API/SharedWorker.

[14] "ServiceWorker API - Web API Interfaces | MDN," Mozilla Foundation, [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/API/Service_Worker_API.

[15] "Google Trends - Ενδιαφέρον για Αναζήτηση Ιστού: adobe flex, microsoft silverlight, java applet, html5, gwt - Παγκοσμίως, 2004 - παρόν," [Online]. Available:

http://www.google.com/trends/explore#q=adobe++flex,microsoft+silverlight,java++applet,HTML5,gwt.

[16] "Google Trends - Ενδιαφέρον για Αναζήτηση Ιστού: adobe flex, extjs, gwt, vaadin - Παγκοσμίως, 2004 - παρόν," [Online]. Available: http://www.google.com/trends/explore#q=adobe++flex,extjs,gwt,vaadin.

[17] "Canvas element - Wikipedia, the free encyclopedia," [Online]. Available: https://en.wikipedia.org/wiki/Canvas_element.

[18] "Scalable Vector Graphics - Wikipedia, the free encyclopedia," [Online]. Available: https://en.wikipedia.org/wiki/Scalable_Vector_Graphics.

[19] "The Khronos Group Inc.," [Online]. Available: https://www.khronos.org/.

[20] "OpenGL - The Industry Standard for High Performance Graphics," [Online]. Available: https://www.opengl.org/.

[21] T. Parisi, Programming 3D Applications with HTML5 and WebGL, O'Reilly Media, 2014.

[22] "OpenGL Shading Language - Wikipedia, the free encyclopedia," [Online]. Available: https://en.wikipedia.org/wiki/OpenGL_Shading_Language.

[23] "IETF | Internet Engineering Task Force," [Online]. Available: https://www.ietf.org/.

[24] "World Wide Web Consortium (W3C)," [Online]. Available: https://www.w3.org/.

[25] "SIP: Session Initiation Protocol," [Online]. Available: https://www.ietf.org/rfc/rfc3261.txt.

[26] "The XMPP Standards Foundation," [Online]. Available: https://xmpp.org/.

[27] "The WebSocket API (WebSockets) | MDN," [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/API/WebSockets_API.

[28] "Node.js - Wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/Node.js.

[29] "Node.js," [Online]. Available: https://nodejs.org/en/.

[30] "React (JavaScript library) - Wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/React_(JavaScript_library).

[31] "React - Intoducing JSX," [Online]. Available: https://reactjs.org/docs/introducing-jsx.html.

[32] "React Native - Props," [Online]. Available: https://facebook.github.io/react-native/docs/props.html.

[33] "Redux · A Predictable State Container for JS Apps," [Online]. Available: https://redux.js.org/.

[34] "Redux (JavaScript library) - Wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/Redux_(JavaScript_library).

[35] "three.js - Javascript 3D library," [Online]. Available: https://threejs.org/.

[36] "DICOM Multi-Planar Reconstruction," [Online]. Available: https://help.accusoft.com/ImageGear-Net/v20.4/Windows/HTML/topic825.html.

[37] "GitHub - FNNDSC/ami: AMI Medical Imaging (AMI) JS ToolKit," [Online]. Available: https://github.com/FNNDSC/ami.

[38] "Medical imaging in the browser with the A* Medical Imaging (AMI) toolkit. | ESMRMB Annual Scientific Meeting 2017," [Online]. Available: https://epostersonline.com/esmrmb2017/node/3443.

[39] "GitHub - dataarts/dat.gui: dat.gui is a lightweight controller library for JavaScript," [Online]. Available: https://github.com/dataarts/dat.gui.

[40] "webpack," [Online]. Available: https://webpack.js.org/.

[41] "Babel · The compiler for next generation JavaScript," [Online]. Available: https://babeljs.io/.

[42] "Rendering Settings - Google Chrome," [Online]. Available: https://developer.chrome.com/devtools/docs/rendering-settings.

[43] "Volume rendering - Wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/Volume_rendering.

[44] "Advanced Topics > 3D Volume Rendering (VR)," [Online]. Available: https://www.radiantviewer.com/dicom-viewer-manual/3d-volume-rendering.htm.

[45] "GitHub - xtk/X: The X Toolkit," [Online]. Available: https://github.com/xtk/X.

[46] "vtk.js," [Online]. Available: https://kitware.github.io/vtk-js/.

[47] "Home Page - Kitware, Inc.," [Online]. Available: https://www.kitware.com/.

[48] C. Andrikos, G. Rassias, P. Tsanakas and I. Maglogiannis, Real-Time Medical Collaboration Services over the Web, IEEE, 2015.

[49] M. Saleem, M. R. Kamdar, A. Iqbal, S. Sampath, H. F. Deus and A.-C. N. Ngomo, "Big linked cancer data: Integrating linked TCGA and PubMed," in *Web Semantics: Science, Services and Agents on the World Wide Web*.

[50] "Chromium Blog: The future of O3D," [Online]. Available: https://blog.chromium.org/2010/05/future-of-o3d.html.

[51] "https://en.wikipedia.org/wiki/Redux_(JavaScript_library)," [Online]. Available: https://en.wikipedia.org/wiki/Redux_(JavaScript_library).

[52] "vtk.js," [Online]. Available: https://kitware.github.io/vtk-js/.