MPAndroidChart

Έγγραφο Ελέγχου Ποιότητας

Δαμάσκος Σεραφείμ, mai24011

Μπάης Ευάγγελος, mai24032

Τσαούσης Άγγελος, mai25055

Χρυσοχοΐδης Αναστάσιος, mai25067

19 Δεκ 202

Πίνακας Περιεχομένων

1. Εισαγωγή

1.1 Περιγραφή Λογισμικού

Το MPAndroidChart είναι μια δημοφιλής βιβλιοθήκη ανοιχτού κώδικα για την ανάπτυξη γραφημάτων σε εφαρμογές Android. Είναι γραμμένη σε Java και παρέχει τη δυνατότητα δημιουργίας διαφόρων τύπων γραφημάτων, όπως γραμμικά, ραβδογράμματα, πίτες, ραδαρικά, και πολλά άλλα. Ξεχωρίζει για την ευκολία ενσωμάτωσης, την υψηλή παραμετροποίηση και τις πλούσιες λειτουργίες, όπως ζουμ, κύλιση, και διαδραστικότητα. Είναι ιδανική για προγραμματιστές που θέλουν να προσθέσουν οπτικά ελκυστικά και λειτουργικά γραφήματα στις εφαρμογές τους.

1.2 Αναφορές

Όλες οι εκδόσεις της βιβλιοθήκης βρίσκονται στο ακόλουθο gituhub repo : https://github.com/PhilJay/MPAndroidChart . Κατα την διάρκεια σύνταξης της παρούσας αναφοράς το project έχει 44 tags και 2070 commits.

1.3 Επισκόπηση

Η παρούσα ανάλυση χωρίζεται στην Αναφορά Εξέλιξης Ποιότητας και στην Αναγνώριση Προβλημάτων Ποιότητας. Στην Αναφορά Εξέλιξης Ποιότητας αναλύεται η μεθοδολογία που ακολουθήθηκε, αναφέροντας τις εκδόσεις του project που τέθηκαν υπό ανάλυση. Στην συνέχεια παρουσιάζονται τα ευρήματα σε μορφή γραφικών παραστάσεων με την ερμηνεία τους. Αντίστοιχα, στην Αναγνώριση Προβλημάτων Ποιότητας, αναλύεται η μεθοδολογία

2. Αναφορά Εξέλιξης Ποιότητας

2.1 Μεθοδολογία

Η ανάλυση μετρικών έχει σκοπό να παρουσιάσει την εξέλιξη του λογισμικού όσον αφορά την πολυπλοκότητα, την σύζευξη, την συνοχή, την κληρονομικότητα και του μεγέθους. Έτσι, οι μετρικές που λήφθηκαν υπόψη είναι οι Number of Classes (NOC), Depth of Inheritance Tree (DIT), Number of Children Classes (NOCC), Coupling Between Objects (CBO), Lack of Cohesion of Methods (LCOM), Cyclomatic Complexity (CC), Lines of Code (LoC). Στο σύνολο αναλύθηκαν 8 εκδόσεις του λογισμικού υπό εξέταση, οι 2.1.0, 2.1.4, 2.2.0, 2.2.5, 3.0.3, 3.1.0 .

Η ανάλυση πραγματοποιήθηκε με την χρήση του λογισμικού MetricsCalculator, το οποίο παράγει ένα csv αρχείο με τις κλάσεις του project και μετρικές ποιότητας. Από αυτές κρατήθηκαν οι μετρικές που αναφέραμε προηγουμένως.

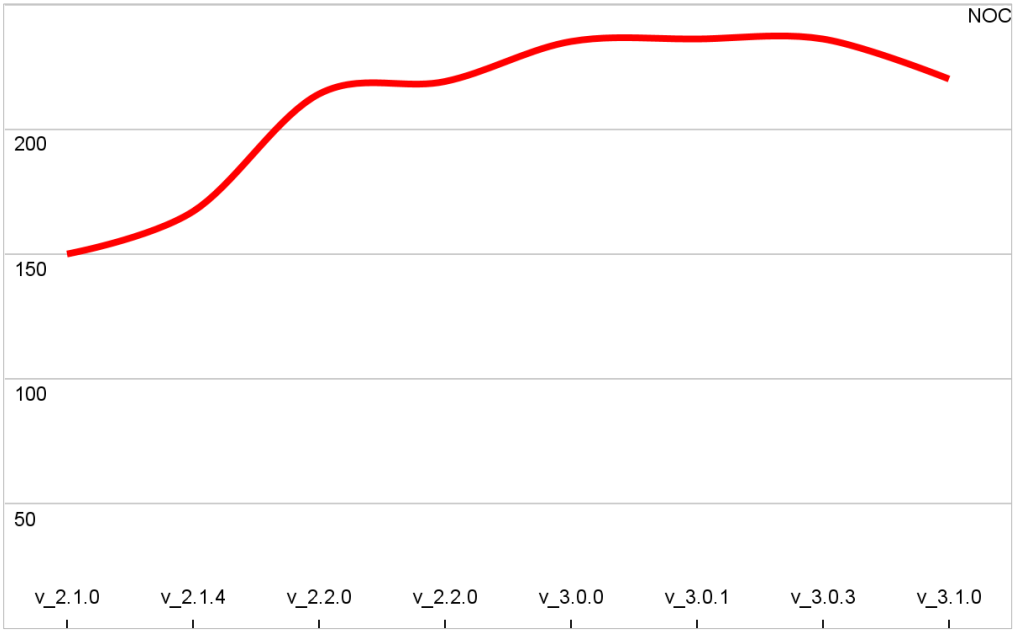
Στο συνοδευόμενο αρχείο excel MPAndroidChart Evolution Analysis βρίσκονται τα αποτελέσματα ανάλυσης για κάθε έκδοση σε ξεχωριστό φύλλο εργασίας. Οι μετρικές αφορούν κάθε κλάση που απαρτίζει μια έκδοση του project και στο τέλος κρατήσαμε ένα πινακάκι για κάθε μετρική που εξετάζουμε, που αναγράφει το σύνολο (SUM) , τον Μέσο Όρο (AVG) και την μέγιστη τιμή (MAX). Στο τελευταίο φύλλο εργασίας υπάρχει συγκεντρωτικός πίνακας (Πίνακας 1) για την σύγκριση των μεγεθών των μετρικών για κάθε έκδοση.

2.2 Ευρήματα

MPAndroid Evolution Analysis									
		v_2.1.0	v_2.1.4	v_2.2.0	v_2.2.5	v_3.0.0	v_3.0.1	v_3.0.3	v_3.1.0
NOC		146	167	214	219	235	236	236	220
DIT	SUM	112	138	350	355	281	282	282	304
	AVG	0.7671232877	0.8263473054	1.635514019	1.621004566	1.2	1.2	1.2	1.381818182
	MAX	4	4	14	14	14	14	14	14
NOCC	SUM	103	113	154	158	161	161	161	162
	AVG	0.7054794521	0.6766467066	0.7196261682	0.7214611872	0.69	0.69	0.685106383	0.7363636364
	MAX	25	27	29	31	34	34	34	32
CBO	SUM	807	923	1041	1100	1249	1252	1270	1245
	AVG	5.52739726	5.526946108	4.864485981	5.02283105	5.34	5.33	5.404255319	5.659090909
	MAX	29	25	24	23	24	24	24	24
LCOM	SUM	16854	19537	21636	26719	28671	29009	30115	30940
	AVG	115.4383562	116.988024	101.1028037	122.0045662	122.53	123.44	128.1489362	140.6363636
	MAX	3916	4560	4950	5460	5995	5995	5995	5886
WMC*(CC)	SUM	266.5594994	307.1414585	343.0099709	358.7994632	377.39	380.38	385.6326904	361.1140889
	AVG	1.825749996	1.83917041	1.602850331	1.638353713	1.61	1.62	1.640990172	1.641427677
	MAX	15	15	15	15	15	15	15	9.833333015
LoC(SIZE 1)	SUM	17920	19372	21539	24001	25157	25139	25790	24310
	AVG	122.739726	116	100.6495327	109.5936073	107.51	106.97	109.7446809	110.5
	MAX	1250	1250	1250	1250	1250	1250	1250	916

Στον ανωτέρω πίνακα παρουσιάζονται αναλυτικά όλα τα αποτελέσματα των εκδόσεων που επιλέξαμε να αναλύσουμε. Στις επόμενες ενότητες ακολουθεί ανάλυση για την κάθε μετρική, καθώς και σχολιασμός.

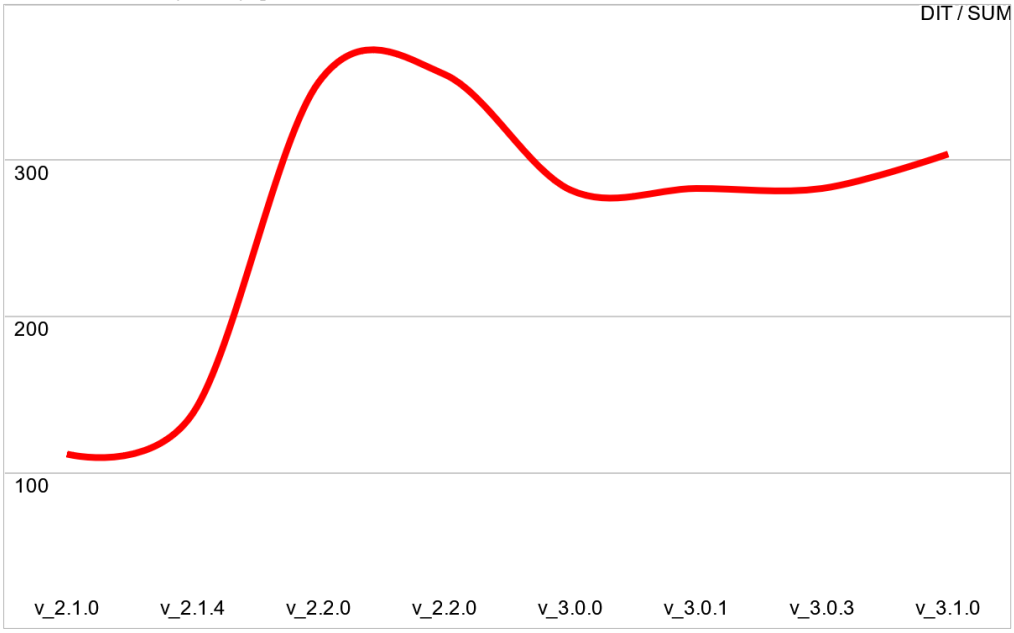
2.2.1 Εξέλιξη Αριθμού Κλάσεων



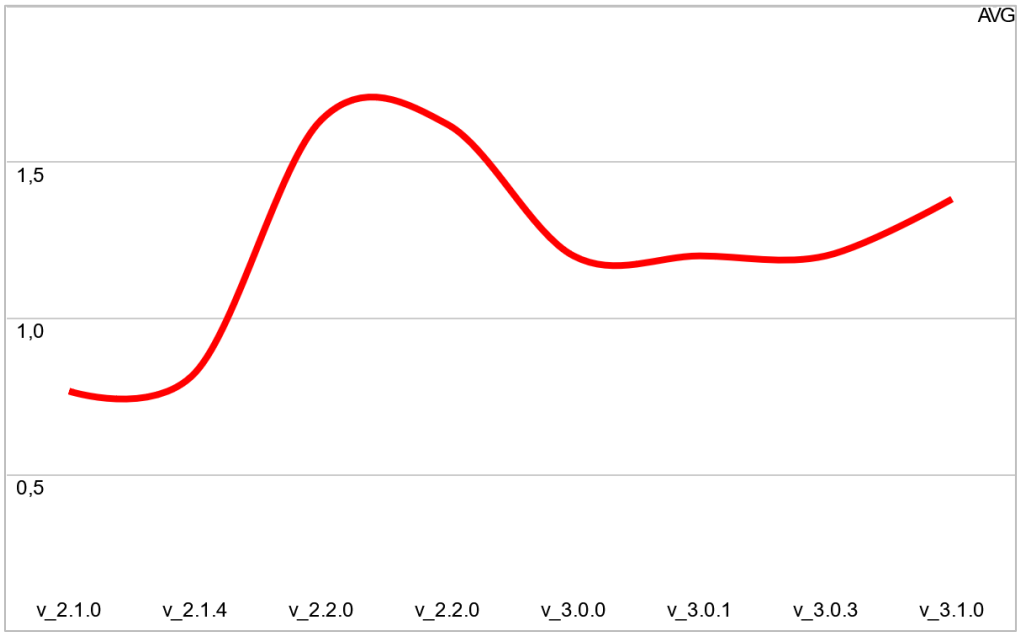
Διάγραμμα 1- NoC

Το γράφημα απεικονίζει την εξέλιξη του **αριθμού κλάσεων.** Η καμπύλη δείχνει ότι το λογισμικό αρχικά αναπτύχθηκε με γρήγορο ρυθμό (αύξηση NOC) και στη συνέχεια σταθεροποιήθηκε. Η μικρή μείωση προς το τέλος μπορεί να υποδεικνύει **βελτιστοποίηση** ή **αφαίρεση περιττών κλάσεων**.

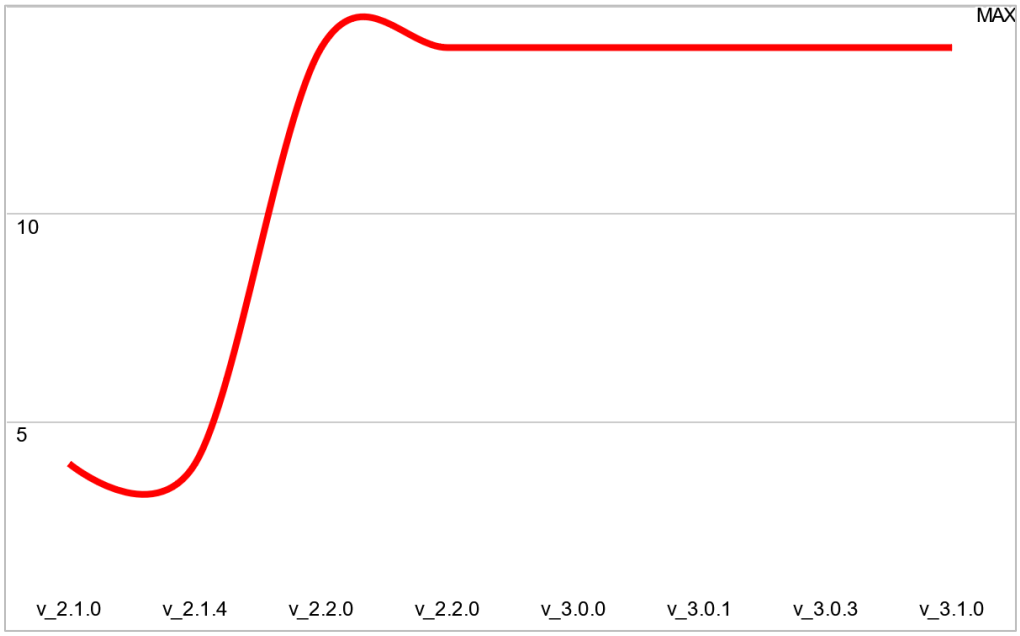
2.2.2 Εξέλιξη DIT



Διάγραμμα 2 - DIT/SUM



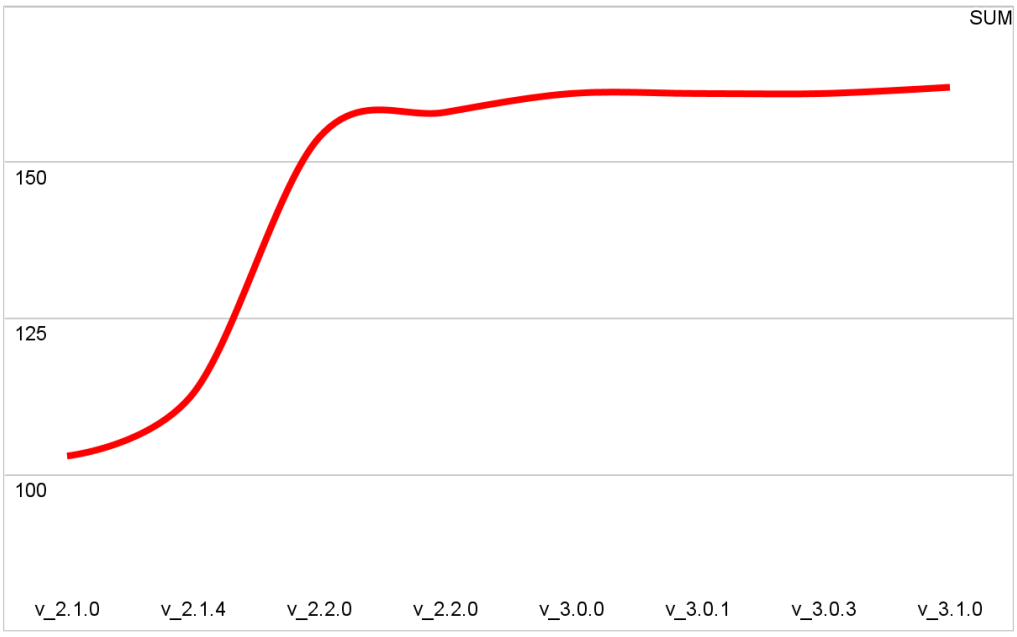
Διάγραμμα 3 - DIT/AVERAGE



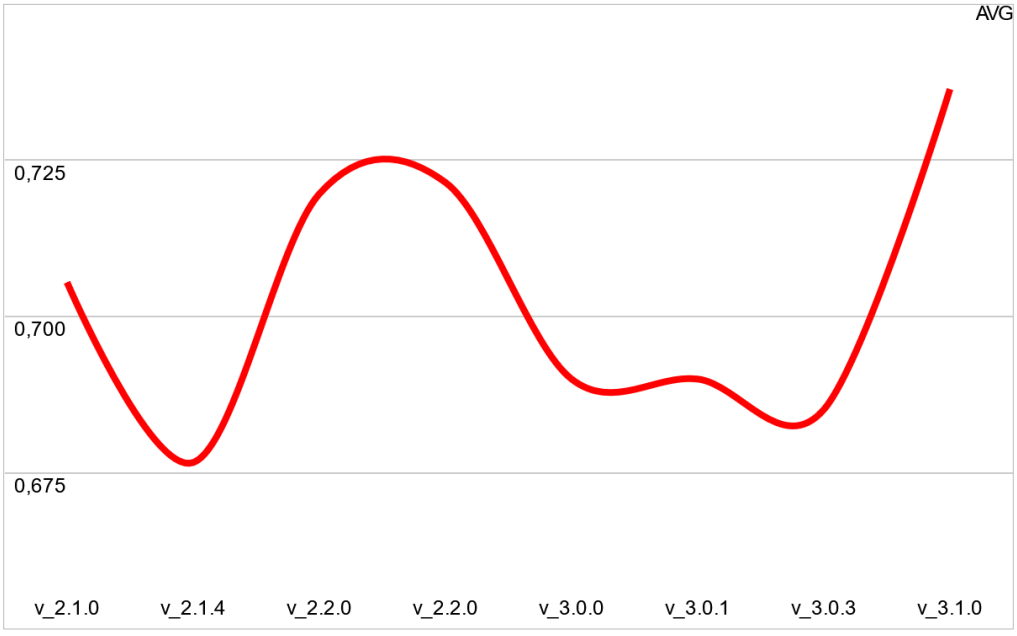
Διάγραμμα 4 - DIT/MAX

Παραπάνω απεικονίζεται η εξέλιξης του βάθους των κλάσεων (DIT). Όπως φαίνεται, λόγω της γρήγορης ανάπτυξης του λογισμικού αρχικά, υπήρξε μια έκρηξη στο βάθος των κλάσεων και αυτό μπορεί να δικαιολογηθεί και μέσω του αριθμού των κλάσεων. Καθώς και οι κλάσεις αυξήθηκαν στο ίδιο διάστημα

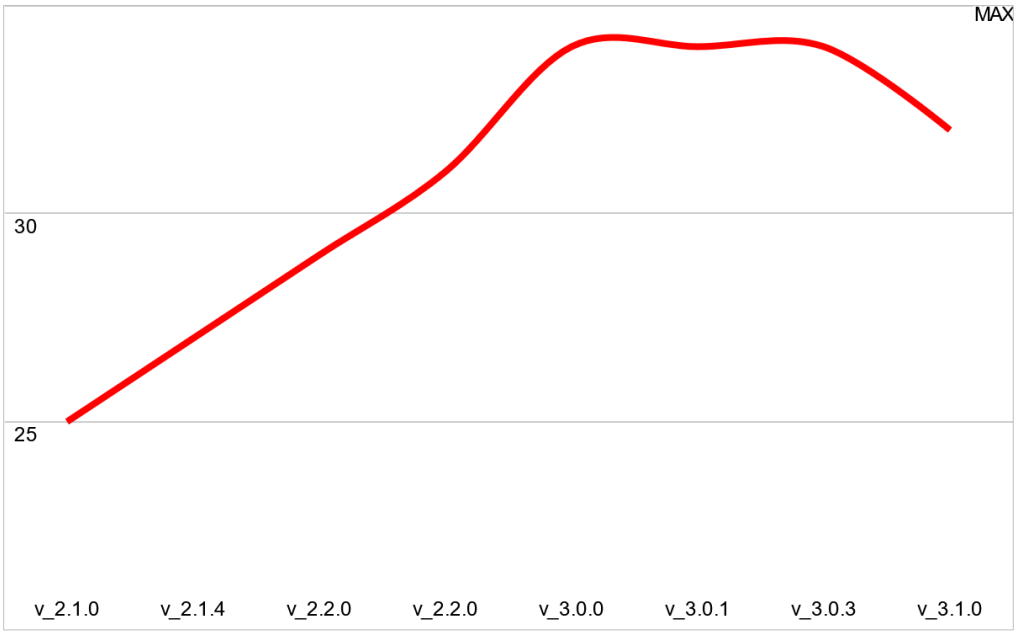
2.2.3 Εξέλιξη NOCC



Διάγραμμα 5 - NOCC/SUM



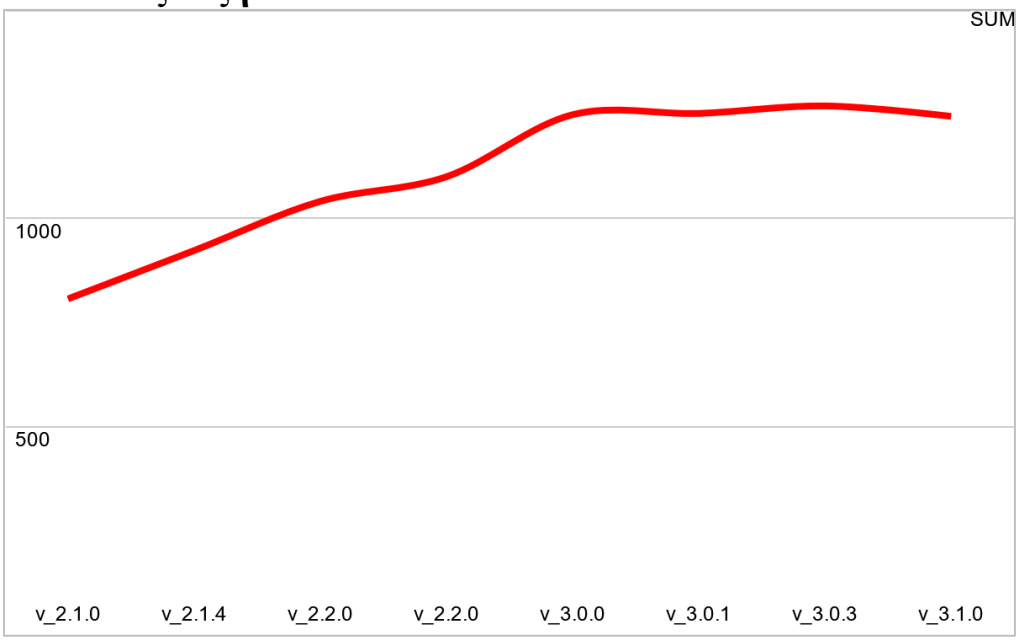
Διάγραμμα 6 - NOCC/AVERAGE



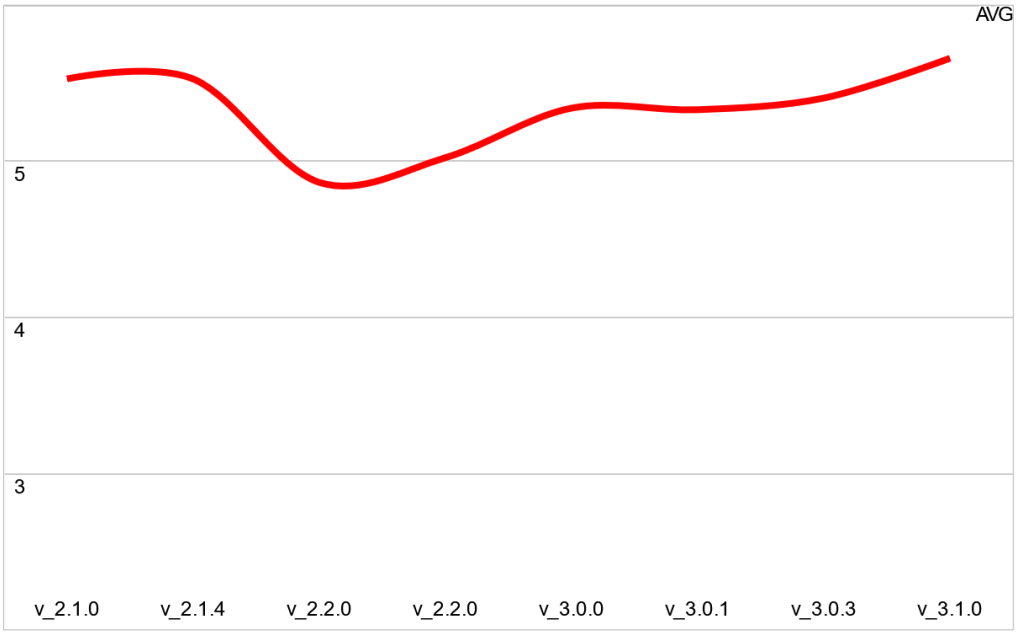
Διάγραμμα 7 - NOCC/MAX

Το συνολικό NOCC παρουσιάζει σημαντική αύξηση στις αρχικές εκδόσεις (v_2.1.0 - v_2.2.0), που υποδηλώνει έντονη χρήση κληρονομικότητας και δημιουργία υποκλάσεων. Στη συνέχεια, ο αριθμός των υποκλάσεων σταθεροποιείται, κάτι που μπορεί να δείχνει ωρίμανση της αρχιτεκτονικής του λογισμικού, με περιορισμένες νέες προσθήκες υποκλάσεων στις νεότερες εκδόσεις.

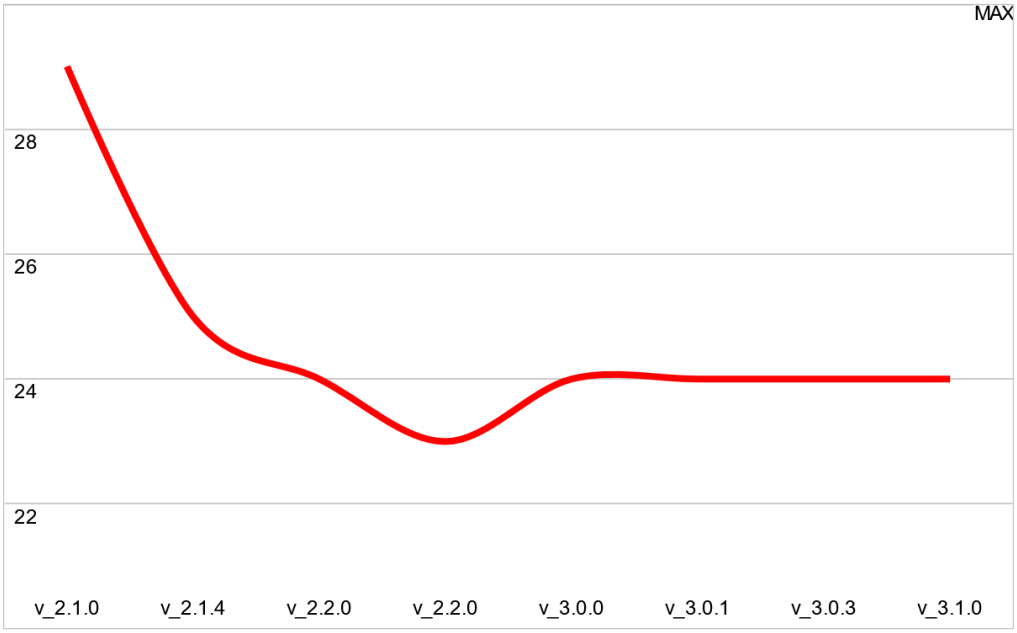
2.2.4 Εξέλιξη CBO



Διάγραμμα 8 - CBO/SUM



Διάγραμμα 9 - CBO/AVERAGE

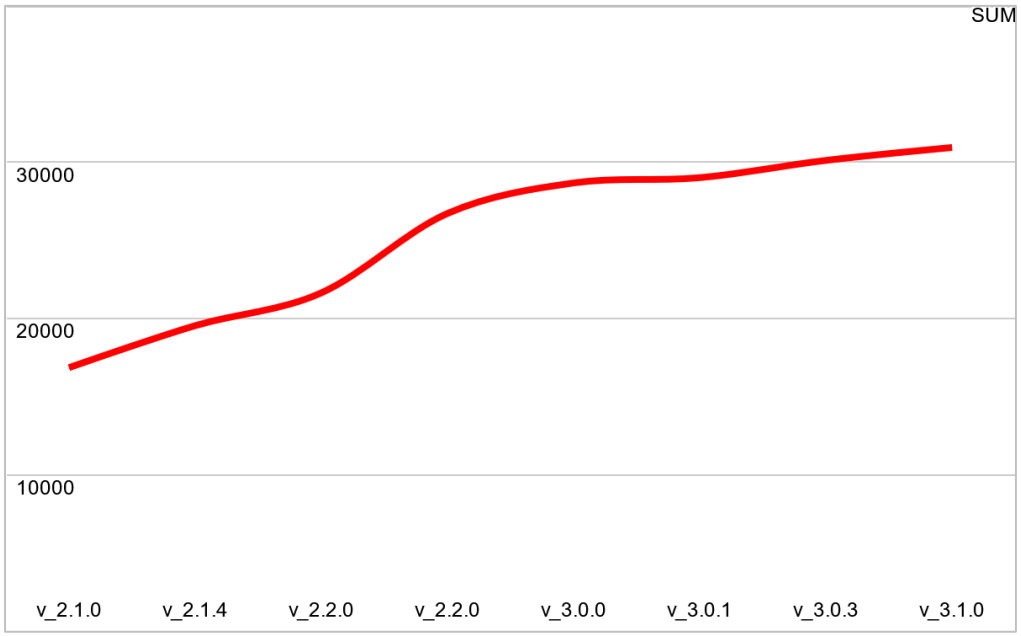


Διάγραμμα 10 - CBO/MAX

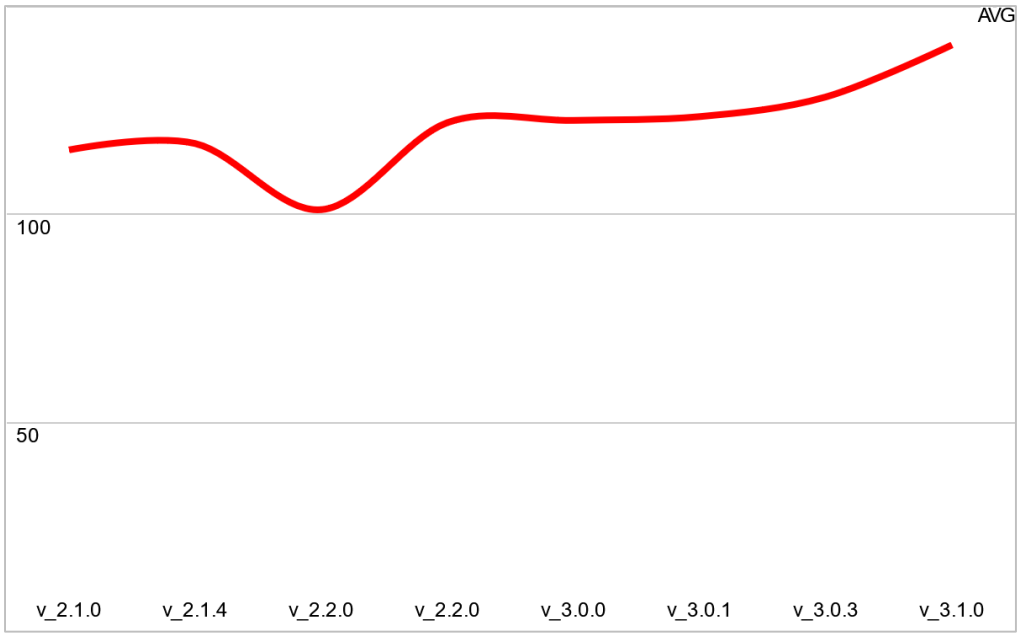
Στην αρχή, το CBO ξεκινά σε υψηλά επίπεδα στην **2.1.0**, παρουσιάζοντας στη συνέχεια πτώση στην έκδοση **2.2.0**, γεγονός που υποδηλώνει μείωση της σύζευξης μεταξύ κλάσεων. Από εκείνο το σημείο και μετά, παρατηρείται μια ανοδική τάση με σταδιακή αύξηση του CBO μέχρι την **3.1.0**, όπου φτάνει στην υψηλότερη τιμή του. Η αύξηση αυτή μπορεί να οφείλεται στην εισαγωγή νέων εξαρτήσεων μεταξύ κλάσεων, ενδεχομένως λόγω προσθήκης νέας λειτουργικότητας ή επεκτάσεων στη σχεδίαση. Συνολικά, η τάση υποδεικνύει ότι, αν και η σύζευξη είχε αρχικά βελτιωθεί, οι νεότερες εκδόσεις παρουσιάζουν υψηλότερη εξάρτηση μεταξύ των κλάσεων, κάτι που μπορεί να επηρεάσει τη

συντηρησιμότητα και την ευελιξία του λογισμικού.

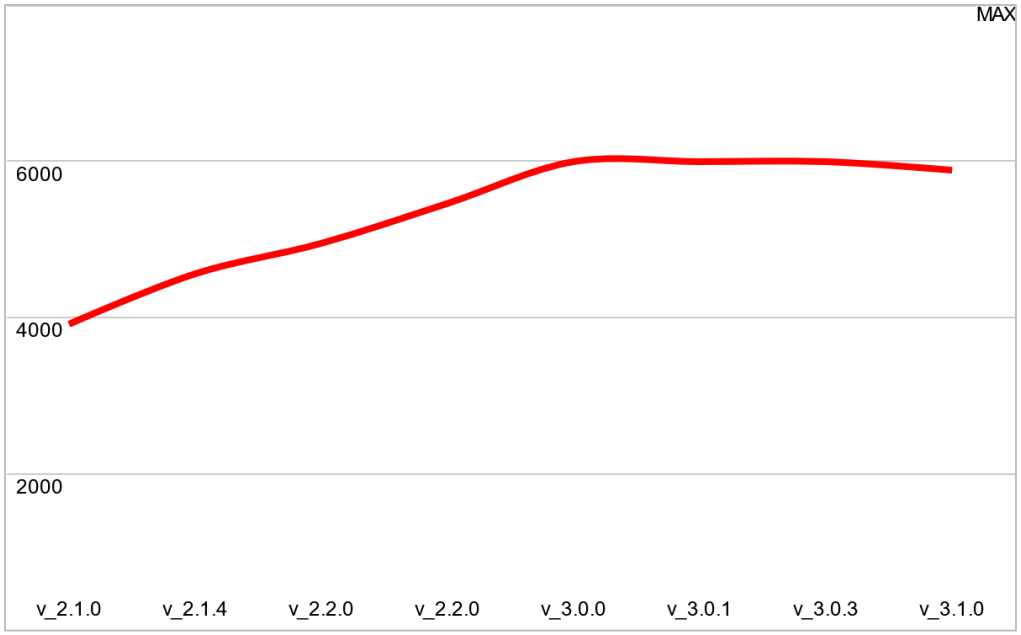
2.2.5 Εξέλιξη LCOM



Διάγραμμα 11 - LCOM/SUM



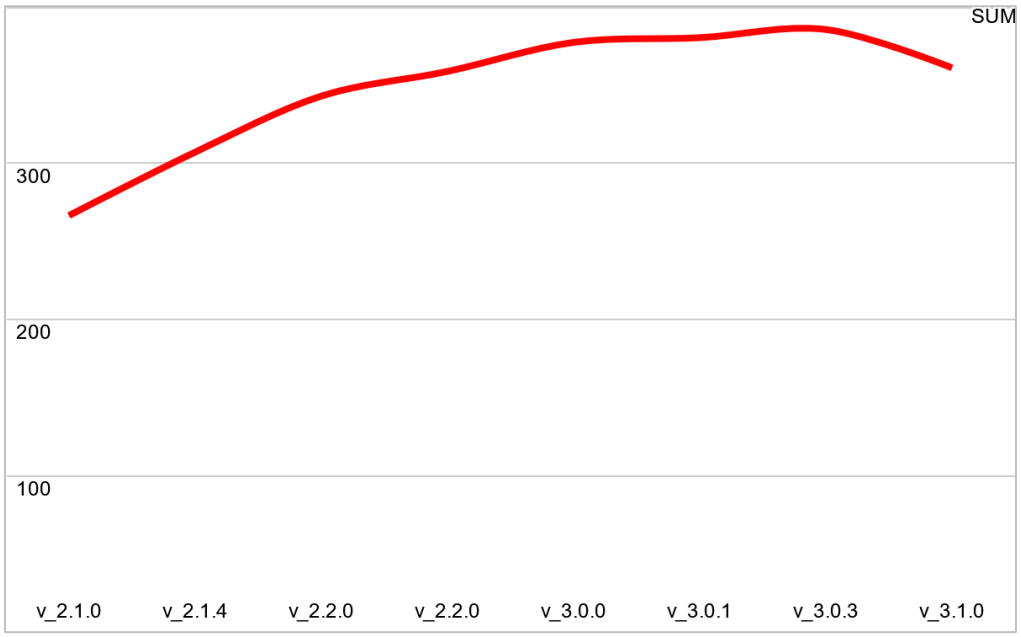
Διάγραμμα 12 - LCOM AVERAGE



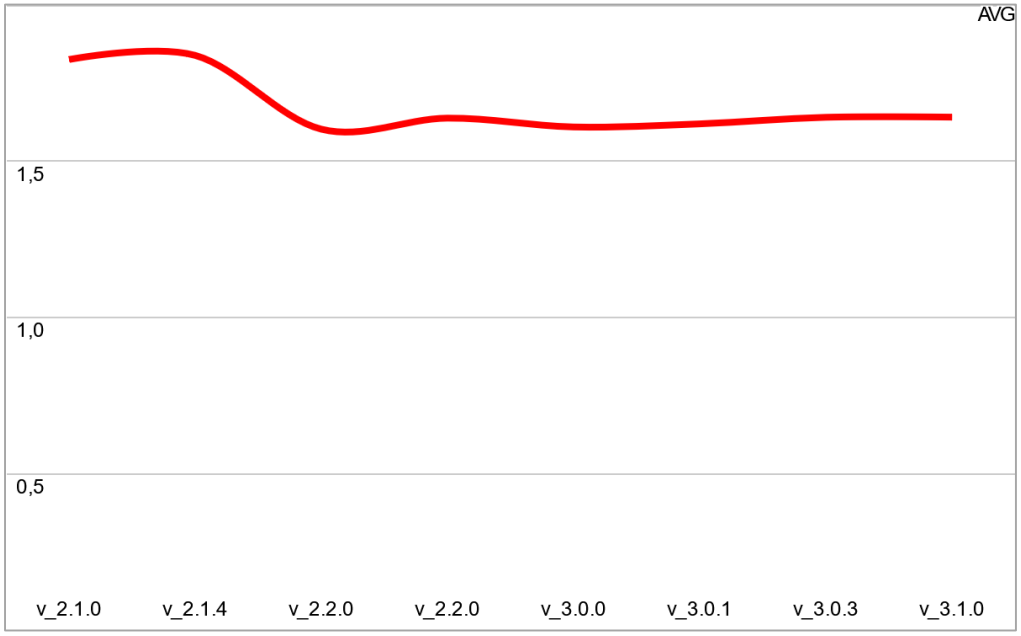
Διάγραμμα 13 - LCOM/MAX

Από τα δεδομένα παρατηρούμε μια γενική αύξηση του δείκτη LCOM από την έκδοση 2.2.0 έως την έκδοση 3.1.0, υποδεικνύοντας μείωση της συνοχής των μεθόδων στις κλάσεις. Η απότομη αύξηση μετά την έκδοση 3.0.0 μπορεί να υποδεικνύει αυξημένη πολυπλοκότητα και δυσκολία στη συντήρηση του λογισμικού.

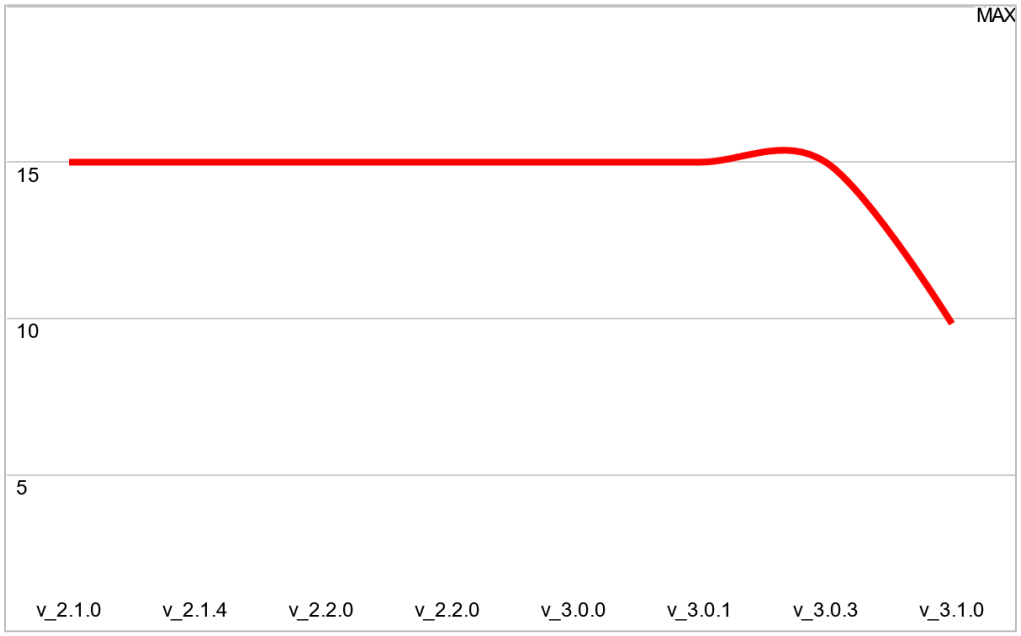
2.2.6 Εξέλιξη WMC*(CC)



Διάγραμμα 14 - CC/SUM



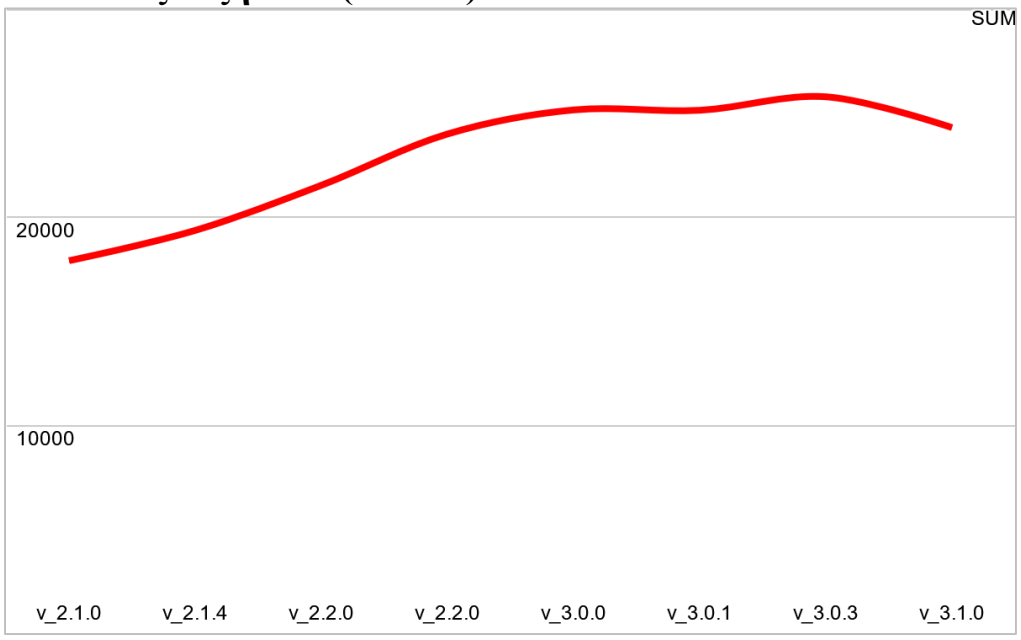
Διάγραμμα 15 - CC/AVERAGE



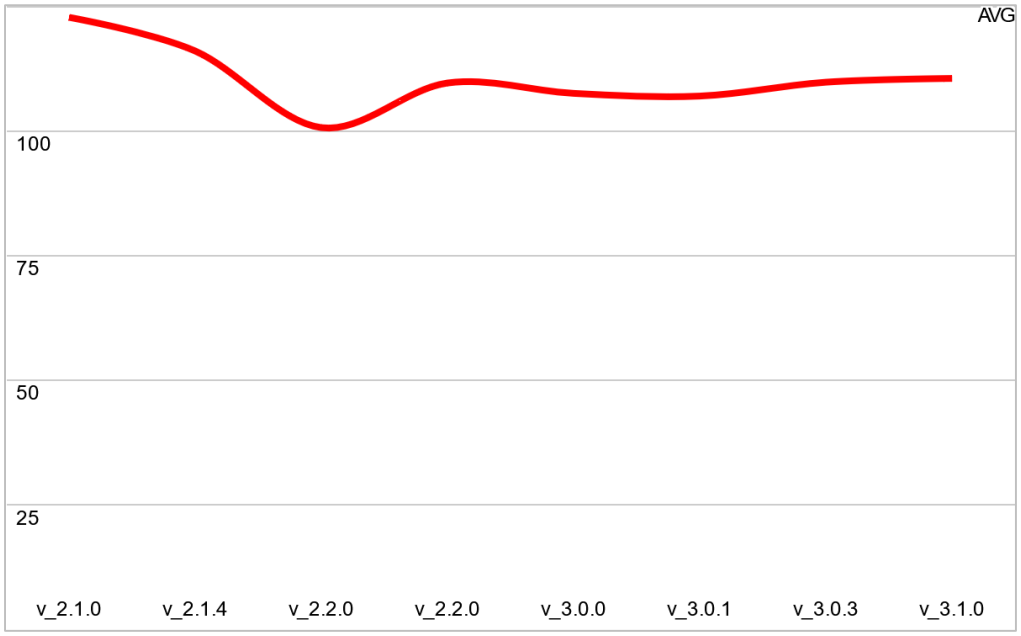
Διάγραμμα 16 - CC/MAX

Η μείωση της κυκλωματικής πολυπλοκότητας στην 3.1.0 δείχνει ότι έγιναν προσπάθειες βελτίωσης του κώδικα, καθιστώντας τον πιο ευανάγνωστο και λιγότερο περίπλοκο σε σχέση με τις προηγούμενες εκδόσεις, κάτι που μπορεί να διευκολύνει τη συντήρηση και την επέκταση του λογισμικού.

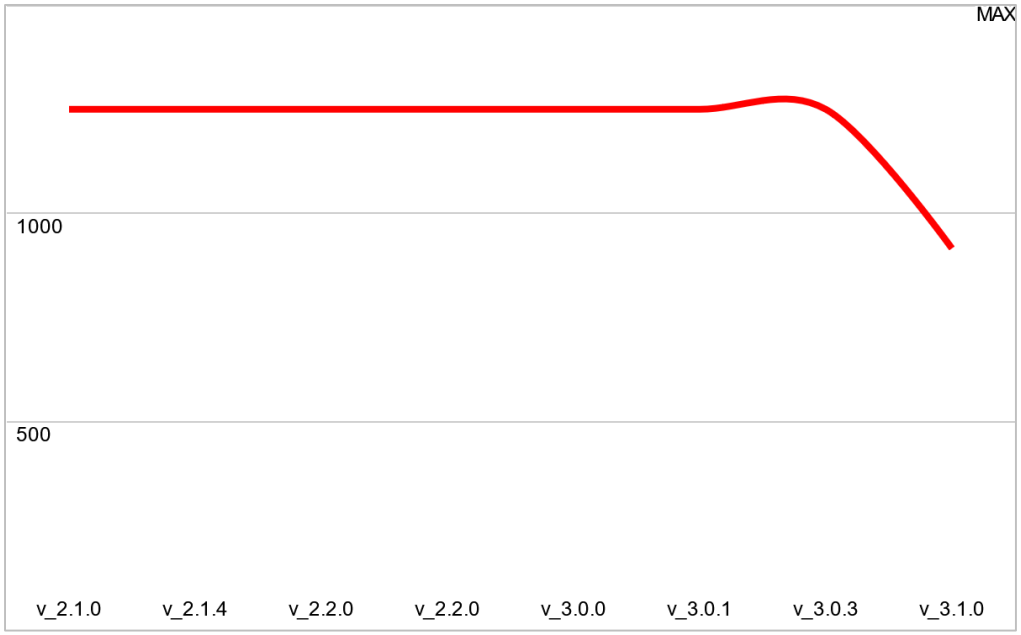
2.2.7 Εξέλιξη LoC (SIZE 1)



Διάγραμμα 17 - LOC/SUM



Διάγραμμα 18 - LOC/AVERAGE



Διάγραμμα 19 - LOC/MAX

Παρομοίως και στην ανάλυση της μετρικής LoC (Size 1) παρατηρείται ότι στην τελευταία έκδοση έχουν γίνει προσπάθειες βελτίωσης του κώδικα.

2.3 Παρατηρήσεις

Παρατηρείται ότι η μεγαλύτερη κλάση για όλες τις εκδόσεις από άποψη LoC έχει ελλατωθεί σημαντικά στην τελευταία έκδοση ή έχει διαγραφεί. Παρατηρήθηκε ότι κατά την εξέλιξη του project.

3. Αναγνώριση Προβλημάτων Ποιότητας

3.3 Μεθοδολογία

Σε ξεχωριστό αρχείο εξετάσαμε τις προβληματικές κλάσεις στην τελευταία έκδοση του project (v.3.1.0). Για την ανάλυση χρησιμοποιήθηκε το ίδιο εργαλείο (Metrics Calculator.jar) κρατώντας τις μετρικές CBO, LCOM, WMC*, LoC. Για κάθε μετρική επισημάνθηκε με κίτρινη σκίαση το 10% των κλάσεων που έχουν τις μεγαλύτερες τιμές.

Στη συνέχεια, πραγματοποιήθηκε κατηγοριοποίηση των κλάσεων με βάση την προτεραιότητα για αναδόμηση (refactoring). Ανάλογα με το πόσες μετρικές έχει η κλάση που την καθιστούν στο υψηλότερο 10%, δηλαδή πόσες μετρικές της κλάσης έχουν κίτρινη σκίαση, χαρακτηρίζεται η κλάση ως εξής :

- LOW, για 1 μετρική
- MODERATE, για 2 μετρικές
- HIGH, για 3 μετρικές
- VERY HIGH, για 4 μετρικές

Σε περίπτωση που μια μετρική είναι πενταπλάσια του Μέσου Όρου της σε όλο το project, τότε ανεβαίνει σε προτεραιότητα κατα μία βαθμίδα. Για παράδειγμα αν μια κλάση έχει χαρακτηριστεί LOW επειδή η μετρική LCOM βρίσκεται στο υψηλότερο 10% , και ταυτόχρονα είναι πάνω από πενταπλάσια του MO της, τότε η κλάση παίρνει τον χαρακτηρισμό MODERATE.

3.4 Ευρήματα

Στο σύνολο των 220 κλάσεων του project, 27 κλάσεις έχουν χαρακτηριστεί LOW priority, 16 κλάσεις έχουν χαρακτηριστεί MODERATE priority , με 4 από αυτές να έχουν αναβαθμιστεί από LOW, 8 κλάσεις έχουν χαρακτηριστεί HIGH priority, με 4 από αυτές να εχουν αναβαθμιστεί από MODERATE και 5 κλάσεις έχουν χαρακτηριστεί VERY HIGH priority με 4 από αυτές να έχουν αναβαθμιστεί από HIGH.

Προτεραιότητα	Αριθμός κλάσεων	Κλάσεις που αναβαθμίστηκαν	Επίπεδο Αναβάθμισης
LOW	27	0	
MODERATE	16	4	LOW → MODERATE
HIGH	8	4	MODERATE → HIGH
VERY HIGH	5	4	HIGH → VERY HIGH
NO PRIORITY	164	0	

3.5 Παρατηρήσεις

Παρατηρείται ότι, πέρα από το γεγονός ότι αρκετές κλάσεις που μετρικές τους ανήκουν στο υψηλότερο 10%, έχουν πολλαπλάσια τιμή του Μέσου Όρου τους και ανεβαίνουν σε προτεραιότητα για αναδόμηση. Συγκεκριμένα, κλάσεις όπως η Chart.java και η BarLineChartBase έχουν πολύ υψηλό LCOM που αγγίζει αύξηση κατά 4087,08% και 3708,91% αντίστοιχα, του Μέσου Όρου LCOM του project. Τέτοια αύξηση μπορεί να αποτελεί ένδειξη ότι οι κλάσεις είναι God Classes, κάτι που αποτελεί κακή σχεδίαση και καθιστά τις κλάσεις δύσκολα συντηρίσιμες. Η διόρθωση τέτοιου είδους κλάσεων βοηθά στην εύκολη επαναχρησιμοποίησή τους, καθιστά τον έλεγχο ευκολότερο, λόγω δημιουργίας επιμέρους μικρότερων κλάσεων και βοηθά στην αναγνωσιμότητα του κώδικα καθώς και στην μακροχρόνια συντήρησή του.

Κλάσεις όπως οι PieChartRenderer.java , BarLineChartBase.java , LineChartRenderer.java και Chart.java , έχουν υψηλό δείκτη της μετρικής LoC. Αυτό ίσως δηλώνει ότι οι κλάσεις έχουν πιθανώς πολύπλοκη λογική κάτι που αυξάνει την πολυπλοκότητα, και ίσως έχουν διπλότυπο κώδικα. Η βελτίωση του δείκτη LoC έχει τα ίδια οφέλη με την διόρθωση του δείκτη LCOM, και είναι εξίσου σημαντική για την αύξηση της ποιότητας του project.