Τεχνολογικό Εκπαιδευτικό Ίδρυμα Κρήτης

Σχόλη Τεχνολογικών Εφαρμογών
Τμήμα Εφαρμοσμένης Πληροφορικής και Πολυμέσων

Πτυχιακή Εργασία

Τίτλος: Υλοποίηση πρόσθετων μηχανισμών στα πλαίσια της πλατφόρμας NetBeans με στόχο την υποστήριξη ανάπτυξης διεπαφών για πολλαπλά περιβάλλοντα.

Βλαχάκης Γεώργιος (Α.Μ.: 2115)

Επιβλέπων Καθηγητής: Ακουμιανάκης Δημοσθένης

Ευχαριστίες

Αρχικά θα ήθελα να ευχαριστήσω την οικογένεια μου, για την υποστήριξη που μου παρείχαν προκειμένου να ολοκληρώσω τις σπουδές μου. Επίσης θα ήθελα να ευχαριστήσω τον Δρ. Ακουμιανάκη Δημοσθένη και τον Δρ. Βιδάκη Νίκο που μου έδωσαν την δυνατότητα να εκπονήσω της πτυχιακή μου εργασία καθώς και την πρακτική μου άσκηση στο iSTLab. Τέλος θα ευχαριστώ τον Βελλή Γιώργο, τον Κότσαλη Δημήτρη, το Γιάννη Μηλολιδάκη και τον Γιώργο Κτιστάκη για την βοήθειά τους.

Abstract

This work is concerned with interactive software design for multiple environments, taking into account recent progress in the field of user interface software and technology. To this end, we take advantage of the model-based UI engineering paradigm, rather than the prominent toolkit-based, since the former provides native predilection towards addressing the diversity of requirements imposed by the wide diversity of devices, users and contexts of use, which all need to be taken take into account when designing for multiple environments.

In such settings of particular importance is the UI design which turns out to be a quite challenging task. Prominent design tools are either biased towards facilitating a rather static context of use, or limited to work as proof of concept tools thus most often supporting only an indicative subset of the interaction facilities provided by the target UIDL specification they implement.

Our approach addresses this limitation aiming to provide a fully functional, professional-level and integrated design environment (IDE) based on the NetBeans platform with full implementation of the UsiXML language specification. The produced tool facilitates the design phase in that it relieves / decouples designers from the burden of understanding unnecessary technical details, imposed by the heterogeneities of target platforms, thus allowing them to exclusively focus on the UI design. A key property of the tool is that it integrates custom software components that allow run-time translation of the actions designers take during a UI design phase into a UsiXML specification reflecting each time visually the designed outcome.

Keywords: UI design for multiple environments, Model-Based UI development, Model-based UI engineering, UsiXML, NetBeansIDE, NetBeans platform, modules development.

Σύνοψη

Τα τελευταία κυρίως χρόνια το τεχνολογικό σκηνικό έχει εξελιχθεί και συνεχίζει να εξελίσσεται άρδην γεγονός που καταμαρτυρείται από την εισβολή και εκτεταμένη διαθεσιμότητα μιας ευρείας γκάμας καινοφανών υπολογιστικών συσκευών, βρίσκοντας ολοένα και περισσότερους χρήστες να αλληλεπιδρούν, ως μέρος των καθημερινών τους πρακτικών, με πλήθος εφαρμογών υπό μια ευρεία γκάμα διαφορετικών συνθηκών τόσο σε επίπεδο πλατφόρμας όσο και σε επίπεδο περιβάλλοντος χρήσης γενικότερα. Ως εκ τούτου, μια σημαντική παράμετρος που αξίζει να συλλογιστούμε σε τέτοιου είδους περιβάλλοντα πολλαπλής χρήσης (‘multi-environment’) σχετίζεται με τη δυνατότητα σχεδίασης, και υπολογιστικής υλοποίησης διεπαφών ικανών να εκτελεσθούν και να ανταποκρίνονται στους περιορισμούς / δυνατότητες καθενός από αυτά λαμβάνοντας υπόψη τις εκάστοτε ιδιομορφίες. Ωστόσο η ικανότητα προσαρμογής και εκτέλεσης μιας διεπαφής δεν πρέπει να θεωρείται ούτε απλή υπόθεση ούτε γηγενώς εκχωρηθείσα, ως δυνητικώς υποστηριζόμενο ποιοτικό χαρακτηριστικό κάποιας διαθέσιμης και δημοφιλούς γραφικής εργαλειοθήκης.

Η αδυναμία αυτή αποδίδεται κυρίως στη συμβατική μέθοδο που ακολουθείται για την ανάπτυξη διεπαφών, δηλαδή στον χαμηλού επιπέδου προγραμματισμό βασισμένο σε εργαλειοθήκες (toolkit-based programming), καθιστώντας υποχρεωτική την ανάπτυξη κατά-περίπτωση (adhoc) διεπαφών για κάθε μια από τις υποστηριζόμενες πλατφόρμες. Καθίσταται λοιπόν προφανής η ανάγκη γνώσης και χαμηλού προγραμματισμού διεπαφών σε τόσες γλώσσες προγραμματισμού ή εργαλειοθήκες όσα και τα διαφορετικά υποστηριζόμενα περιβάλλοντα. Η κατάσταση περιπλέκεται ακόμη περισσότερο λαμβάνοντας υπόψη την ανάγκη μετάφρασης και ενσωμάτωσης δυνητικώς αναδυόμενων απαιτήσεων στα πλαίσια υπαρχόντων διεπαφών-συστημάτων.

Σε αυτά τα πλαίσια η παρούσα εργασία στοχεύει να συνεισφέρει προσεγγίζοντας το παραπάνω πρόβλημα, αυτό της ανάπτυξης διεπαφών για πολλαπλά περιβάλλοντα, μέσω της ανάδειξης και επιλογής εναλλακτικών μεθόδων περιγραφής διεπαφών προτείνοντας και υποβοηθώντας τη διαδικασία σχεδίασης αυτών μέσω ενός εξειδικευμένου και εύχρηστου ολοκληρωμένου συστήματος ανάπτυξης (IDE – Integrated Development Environment). Συγκεκριμένα, η παρούσα εργασία προτείνει και στοχεύει στην ανάπτυξη ενός πρόσθετου συστατικού (plugin) εκμεταλλευόμενο τα πλεονεκτήματα που ήδη προσφέρει η πολύ δημοφιλής πλατφόρμα NetBeans (NetBeans-platform), με στόχο την υποστήριξη σχεδίασης και περιγραφής διεπαφών για πολλαπλά περιβάλλοντα στόχους (target-environments) μέσω ενός εξειδικευμένου, ευρέως αποδεκτού επαγγελματικού περιβάλλοντος, αυτό του NetBeans (NetBeans IDE).

Λέξεις κλειδιά: Σχεδίαση διεπαφών για πολλαπλά περιβάλλοντα, Μοντελοκεντρική ανάπτυξη διεπαφών, Μοντελοκεντρική μηχανική Διεπαφών, UsiXML, NetBeansIDE, NetBeans platform, ανάπτυξη modules.

Πίνακας περιεχομένων

Πίνακας Εικόνων

1. Εισαγωγή

Στα πλαίσια της αυξητικής τάσης που παρατηρείται στη χρήση νέων υπολογιστικών συσκευών από τους χρήστες προκειμένου να εκτελούνται πλήθος καθημερινών καθηκόντων, είναι ιδιαίτερα σημαντικό να υποστηρίζεται η σχεδίαση και κατασκευή διεπαφών που να είναι ικανές να ανταποκριθούν και να προσαρμοστούν στις διαφορετικές παραμέτρους και ιδιαίτερα χαρακτηριστικά ή/και περιβάλλοντα χρήσης των συσκευών αυτών. Δύο είναι οι κυρίαρχες μεθοδολογίες που χρησιμοποιούνται για την κατασκευή διεπαφών: η μία βασίζεται στον προγραμματισμό αντικειμένων μιας εργαλειοθήκης ή εργαλειοθηκοκεντρικός (toolkit-based) ενώ ο δεύτερος στηρίζεται στις προδιαγραφές βάση μοντέλων ή αλλιώς μοντελοκεντρικός (model-based). Η κυρίαρχη και συνάμα η αρχαιότερη αλλά και ευρύτερα χρησιμοποιούμενη μέθοδος εκ των δύο είναι η πρώτη και αυτό λόγω των αυξημένων δυνατοτήτων που προσφέρει όσον αφορά την πλήρη εκμετάλλευση των υπολογιστικών δυνατοτήτων της εκάστοτε τελικής πλατφόρμας χρήσης. Ωστόσο υπό το πρίσμα των αναδυόμενων απαιτήσεων όπως αυτές εγείρονται στα πλαίσια πλήθους νέων καινοφανών υπολογιστικών συσκευών ο εργαλειοθηκοκεντρικός προγραμματισμός δείχνει να υποφέρει από πλήθος ένθετων γηγενών μειονεκτημάτων που συνοψίζονται στον κατά κανόνα προσανατολισμό των σε μια και μόνο τελική πλατφόρμα. Το τελευταίο έχει έως τώρα μερικώς αντιμετωπιστεί μέσω αφαιρετικών προσεγγίσεων (abstraction layers) στη βάση των εν δυνάμει πιο δημοφιλών πάντα λειτουργικών συστημάτων. Μια τέτοια προσέγγιση είναι η γλώσσα προγραμματισμού java η οποία είναι η πιο δημοφιλής έως τώρα σε αυτά τα πλαίσια από την πλευρά του toolkit-based προγραμματισμού, ωστόσο υπάρχουν και άλλες. Κύριο μειονέκτημα αυτών είναι το γεγονός ότι δεν καλύπτουν κατά κανόνα ίσως το μεγαλύτερο φάσμα εκ των αναδυόμενων υπολογιστικών συσκευών εστιάζοντας κυρίως σε αφαιρέσεις στα πλαίσια πλατφορμών σταθερών υπολογιστών. Ως εκ τούτου, αναγκαία κρίνεται η χρήση ξεχωριστών εργαλειοθηκών για κάθε ένα από τα νέα υπολογιστικά περιβάλλοντα που είναι να υποστηριχθούν. Ωστόσο δεδομένου του μη κοινού της μεθοδολογίας ανάπτυξης που υιοθετείται για κάθε μια τελική συσκευή τόσο η διαδικασία ανάπτυξης όσο και επανασχεδίασης μέσω τις αναπροσαρμογής σε νέες απαιτήσεις καθίσταται ιδιαιτέρα χρονοβόρα, εξαιρετικά απαιτητική και ασύμφορος υπονομεύοντας ανά πάσα στιγμή τη συνοχή (consistency) των τελικών εναλλακτικών λογισμικών. Στα πλαίσια αυτά νέο νόημα δόθηκε στην ανάπτυξη λογισμικού και δει διεπαφών αυτών, που αποτελεί και το πιο απαιτητικό και κρίσιμο κομμάτι της ανάπτυξης, μέσω μοντέλων ώστε να απομονωθεί ανάλογα με το επίπεδο σχεδίασης το είδος των λεπτομερών που ενδιαφέρουν την εκάστοτε στιγμή. Ενδεικτικές προσεγγίσεις αυτής της κατηγορίας είναι γλώσσες όπως η UsiXML, η Multimodal Teresa αλλά και αρκετές άλλες. Κύριο γνώρισμα αυτών στα πλαίσια της σχεδίασης διεπαφών είναι η απομόνωση μόνο των πιο δημοφιλών διαδραστικών αντικειμένων που απαντώνται στα πιο δημοφιλή περιβάλλοντα χρήσης (δηλ.: κουμπιά, ετικέτες, κα.). Αυτό ερμηνεύεται σε σχέση με τις εργαλειοθηκοκεντρικές μεθόδους ανάπτυξης στο γεγονός ότι δεν εκμεταλλεύονται εν τη γενέσει πλήρως το εύρος των διαθέσιμων διακρατικών δυνατοτήτων των τελικών συσκευών δίνοντας χώρο για περεταίρω βελτιώσεις. Ωστόσο το κυριότερο και αυτό που μας απασχολεί και που επιχειρήθηκε να αντιμετωπιστεί στα πλαίσια αυτής της εργασίας είναι το γεγονός ότι παρατηρείται παντελής έλλειψη ενός αποτελεσματικού και πλήρους λειτουργικότητας εργαλείου (non-research prototype level, non- partial implementations) το οποίο να βοηθά τη φάση σχεδίασης διεπαφών για πολλαπλά περιβάλλοντα. Σε αυτή την κατεύθυνση επιλέξαμε ως γλώσσα αναφοράς τη UsiXML, η λογική της οποίας θα αναλυθεί στη συνέχεια, στα πλαίσια της οποία βασιζόμενοι στην πλατφόρμα NetBeans δημιουργήσαμε ένα ανοιχτό σε επεκτάσεις (modular) εργαλείο ώστε να υποστηρίξουμε

πλήρως τη φάση σχεδίασης διεπαφών στο “Concrete Επίπεδο” (Concrete UI) που είναι και το πιο απαιτητικό και όπου παρατηρείται έλλειψη αντίστοιχων εναλλακτικών. Σε αυτή τη γραμμή το εργαλείο παρέχει πλήρη υλοποίηση και στήριξη των δυνατοτήτων της γλώσσας περιγραφής διεπαφών (UIDL) UsiXML.

Η συνέχεια της εργασίας είναι οργανωμένη ως εξής:

- Στο Κεφάλαιο 2: παρουσιάζονται κύριες έννοιες, και καθιερωμένες μεθοδολογίες ανάπτυξης διεπαφών για πολλαπλά περιβάλλοντα.
- Στο Κεφάλαιο 3:, παρουσιάζονται τα δημοφιλέστερα ολοκληρωμένα περιβάλλοντα ανάπτυξης (IDEs) για την υποστήριξη της φάσης σχεδίασης διεπαφών.
- Στο Κεφάλαιο 4: παρουσιάζεται η οικεία προσέγγιση όσον αφορά τη σχεδίαση διεπαφών για πολλαπλά περιβάλλοντα.
- Στο Κεφάλαιο 5: παρουσιάζεται η υλοποίηση του εργαλείου που αναπτύχθηκε, η αρχιτεκτονική αυτού όπως επίσης λοιπά επιμέρους στοιχεία αυτού ενώ τέλος
- Στο Κεφάλαιο 6: υπάρχει ο επίλογος της πτυχιακής καθώς επίσης μελλοντικές επεκτάσεις για την εξέλιξη του εργαλείου.

2. Σχεδίαση διεπαφών για πολλαπλά περιβάλλοντα.

Τα τελευταία κυρίως χρόνια παρατηρείται ραγδαία ανάπτυξη και διείσδυση σε επίπεδο καθημερινών πρακτικών πολλών νέων τύπων υπολογιστικών συσκευών ικανών να καλύψουν μια ολοένα διευρυνόμενη γκάμα απαιτήσεων συναρτήσει των δυνητικών χρηστών τους. Τέτοιου είδους συσκευές ποικίλουν από κινητά τηλέφωνα, smart phones, PDAs, pocket PCs, Tablet PCs, electronic whiteboards, κοκ. Ωστόσο το εντυπωσιακό είναι ότι δεν πρόκειται μόνο απλά για εναλλακτικούς τύπους εκδοχών της ίδιας (συνήθως εξαιρετικά χαμηλής) υπολογιστικής ισχύος σε σχέση με της καθιερωμένης των σταθερών υπολογιστών, αντιθέτως οι επεξεργαστικές τους ικανότητές είναι εντυπωσιακές προσφέροντας τη δυνατότητα για νέου είδους καινοτομίες και πρωτοποριακές εφαρμογές. Ωστόσο όπως ίσως είναι αναμενόμενο και λογικό αυτό δεν μπορεί να μην συνεπάγεται το αντίστοιχο αντίτιμο όσον αφορά την ικανότητα των καθιερωμένων ενδεδειγμένων πρακτικών να ανταπεξέλθουν στη σχεδίαση συστημάτων για πολλαπλά περιβάλλοντα και κυρίως στη σχεδίαση των διεπαφών αυτών.

Εικόνα 1: Υπολογιστικές συσκευές

Τέτοιου είδους προκλήσεις συνοψίζονται στη:

- *Φάση ανάλυσης και σχεδίασης διεπαφών:* για κάθε ένα υποστηριζόμενο υπολογιστικό πλαίσιο χρήσης (context of use) λαμβάνοντας υπόψιν διαφοροποιήσεις των υπολογιστικών πόρων (μνήμη, οθόνη, πιθανές εισόδους), πλατφόρμας (android, windows, κοκ.), εξωγενών παραγόντων (περιβάλλον χρήσης: τρέχουσα τοποθεσία χρήστη, θορυβώδες ή σκιερό περιβάλλον, κοκ.).
- *Φάση υλοποίησης:* ώστε να υποστηριχθεί κάθε μια υπολογιστική συσκευή, κυρίως όσον αφορά στη διεπαφή αυτών.
- *Επαναμηχανική (re-engineering) του συστήματος:* ώστε να ανταποκριθεί σε αναδυόμενες απαιτήσεις (λειτουργικές και μη) ή στη διόρθωση σφαλμάτων μιας και τα ίδια λάθη δεν είναι αποκλειστικό "δικαίωμα" όλων των εκδοχών της εφαρμογής.
- Στη δυσκολία της εξασφάλισης συνέπειας (consistency) μεταξύ των εναλλακτικών εκδοχών της εφαρμογής.

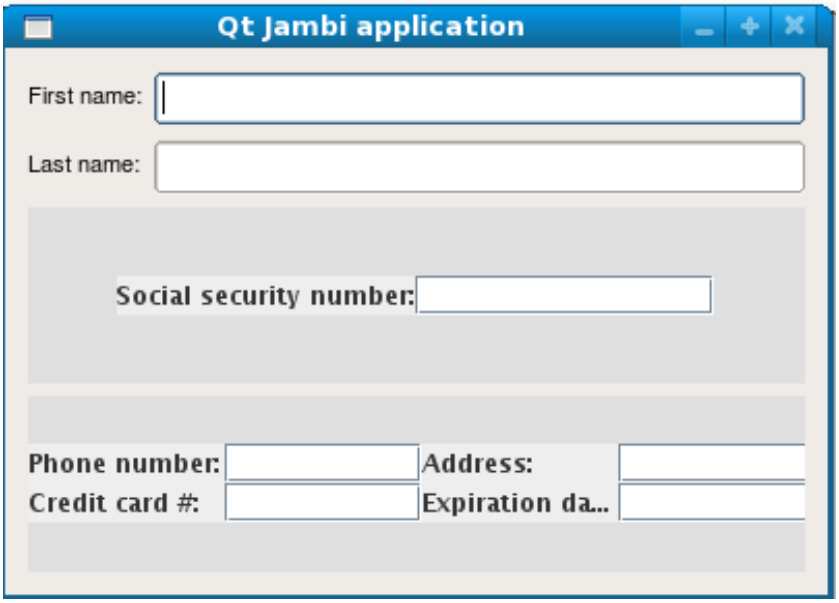
2.1 Κύριες Μέθοδοι Ανάπτυξης Λογισμικού για Πολλαπλά Περιβάλλοντα

Εργαλειοθηκοκεντρικός Προγραμματισμός (Toolkit-Based Programming)

Στους ηλεκτρονικούς υπολογιστές, ένα widget toolkit, widget library, ή GUI toolkit είναι ένα σύνολο από γραφικά αντικείμενα (widgets) τα οποία χρησιμοποιούνται στο σχεδιασμό εφαρμογών με γραφικό περιβάλλον (GUI). Η εργαλειοθήκη από μόνη της είναι ένα κομμάτι του λογισμικού που συνήθως είναι χτισμένο στην κορυφή του λειτουργικού συστήματος, η του window manager, και παρέχει προγράμματα μέσω των (API), που τους επιτρέπουν να κάνουν χρήση των widgets. Κάθε widget διευκολύνει το χρήστη, και εμφανίζεται ως ορατό τμήμα του GUI του υπολογιστή. Ένα Widget set εργαλείων μπορεί να υποστηρίζεται από μια η από περισσότερες πλατφόρμες. Widgets που παρέχονται από ένα σύνολο εργαλείων, συνήθως ακολουθούν ενιαίες προδιαγραφές σχεδιασμού, συμπεριλαμβανομένης της αισθητικής, έτσι ώστε να δώσουν μια αίσθηση συνολικής συνοχής μεταξύ των διαφόρων τμημάτων της εφαρμογής. Επίσης περιλαμβάνουν λογισμικό το οποίο βοηθάει στη δημιουργία των διαχειριστών παραθύρων (windows manager). Κάποια widgets υποστηρίξουν την αλληλεπίδραση με το χρήστη, για παράδειγμα ετικέτες, κουμπιά κ.α. Άλλα λειτουργούν ως containers, στα οποία τοποθετούνται άλλα widget, για παράδειγμα windows, panels, tabs. Οι περισσότερες εμπορικές widget εργαλειοθήκες χρησιμοποιούν event-driving programming, δηλαδή διαχειρίζονται τα events που δημιουργούνται από την αλληλεπίδραση του χρήστη με τα αντικείμενα , για παράδειγμα, όταν ο χρήστης κάνει κλικ σε ένα κουμπί. Όταν ένα συμβάν εντοπιστεί, μεταφέρεται στην εφαρμογή, όπου αντιμετωπίζεται. Ο σχεδιασμός αυτών των εργαλείων έχει επικριθεί γιατί παρουσιάζει υπέρ-απλουστευμένα μοντέλα αλληλεπίδρασης, με αποτέλεσμα οι προγραμματιστές να είναι επιρρεπείς σε λάθη, και να είναι δύσκολο να επεκταθούν σε υπερβολικά πολύπλοκο κώδικα. Με την μεγάλη ποικιλία διαφορετικών υπολογιστικών συσκευών και διαφορετικών πλατφόρμων, εκτός από την αυξημένη δυσκολία στη διαδικασία ανάπτυξης γεννάται και μια μεγάλη ποικιλία απαιτήσεων που πρέπει να ληφθούν υπόψη, ανάλογα με στις δυνατότητες της εκάστοτε συσκευής και το πλαίσιο χρήσης. Επίσης για τους σχεδιαστές και τους προγραμματιστές υπάρχει μια συνεχώς αυξανόμενη δυσκολία σε σχέση με το πλήθος των γλωσσών προγραμματισμού που θα πρέπει να γνωρίζουν. Τέλος αυτή η προσέγγιση κάνει εξαιρετικά δύσκολη τη διαδικασία του ανασχεδιασμού (re-engineering) πράγμα που καθιστά αδύνατη τη παροχή διαφορετικών εκδόσεων της ίδιας διεπαφής. Σε επόμενα κεφάλαια γίνεται μια πιο λεπτομερής αναφορά στις κυριότερες εργαλειοθήκες που χρησιμοποιούνται για τη σχεδίαση διεπαφών.

Εργαλειοθήκη Qt

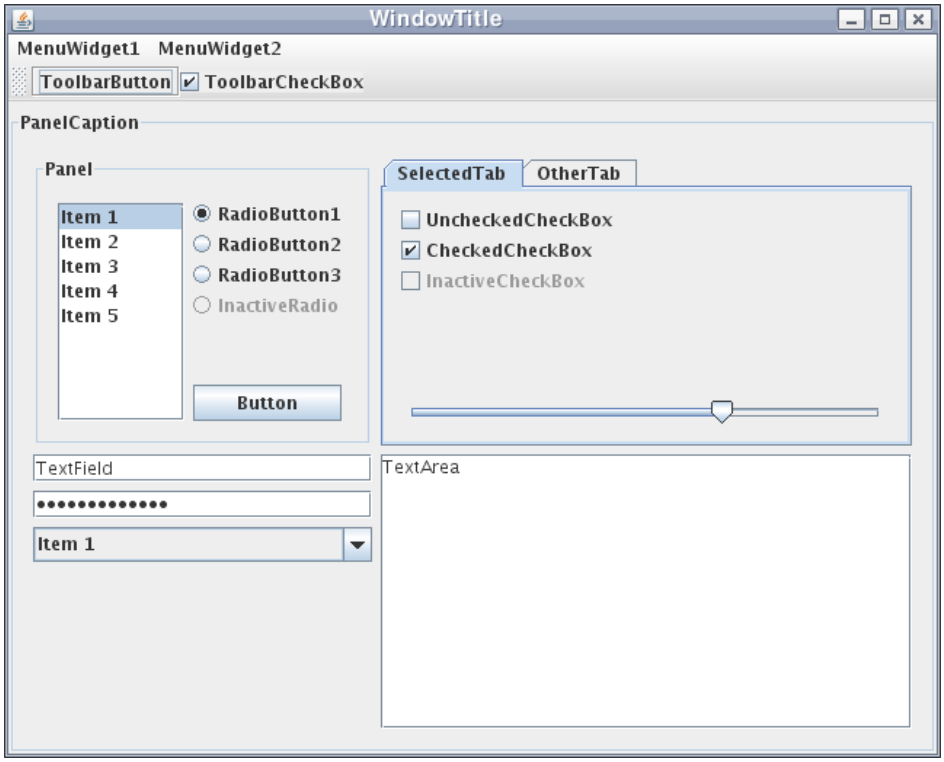
Το Qt είναι ένα widget toolkit που χρησιμοποιείται ευρέως για την ανάπτυξη γραφικών διεπαφών (GUI). Χρησιμοποιείται επίσης για την ανάπτυξη μη-GUI προγραμμάτων όπως εργαλεία γραμμής εντολών και κονσόλες για servers. Χρησιμοποιεί το πρότυπο C++, αλλά κάνει εκτεταμένη χρήση μιας ειδικής γεννήτριας κώδικα Compiler Object Meta (MOC), μαζί με πολλές μακροεντολές για να εμπλουτίσουν τη γλώσσα. Μπορεί επίσης να χρησιμοποιηθεί σε πολλές άλλες γλώσσες προγραμματισμού. Είναι συμβατό με όλες τις κύριες desktop πλατφόρμες (Windows, Mac Os, Linux) και μερικές από τις mobile πλατφόρμες. Στην (Εικόνα 2) υπάρχει μία διεπαφή η οποία βασίζεται στο Qt.



Εικόνα 2: Παράδειγμα διεπαφής βασισμένη στο Qt

Εργαλειοθήκη Swing

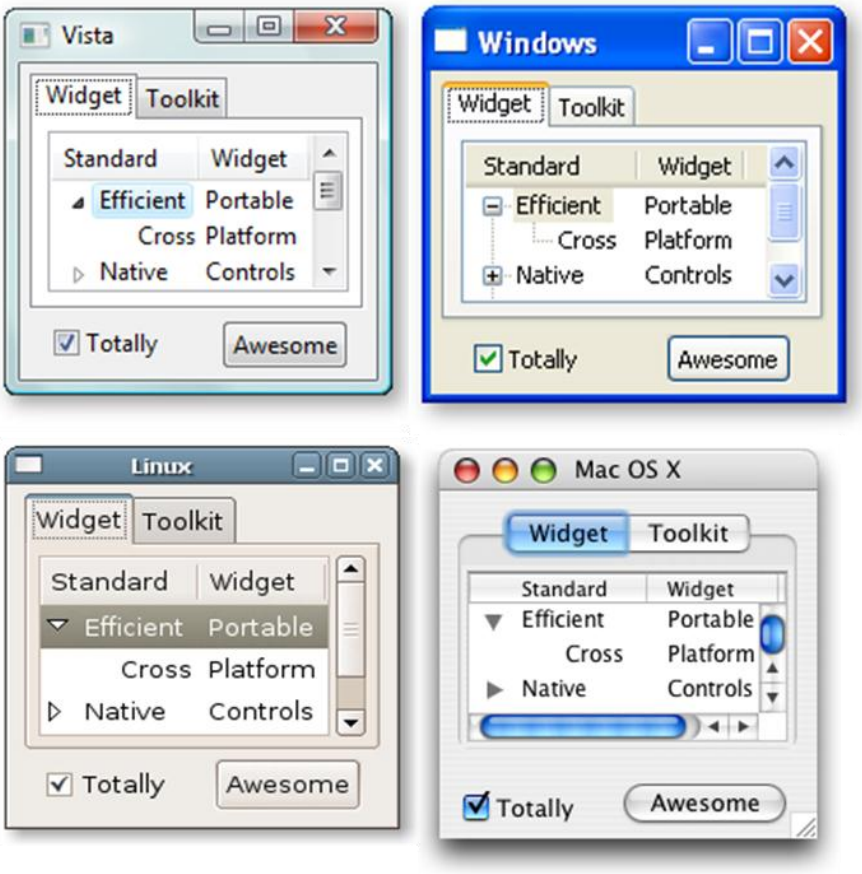
Το Swing είναι ένα Java GUI widget toolkit . Είναι ένα API για την παροχή γραφικών διεπαφών (GUI) για Java εφαρμογές και αποτελεί μέρος των Java Foundation Classes (JFC). Το Swing αναπτύχθηκε για να παρέχει ένα μεγαλύτερο σύνολο GUI αντικειμένων από την προηγούμενη έκδοση AWT (Abstract Window Toolkit), εκτός από τα συνήθη αντικείμενα όπως κουμπιά, combobox, και ετικέτες, το Swing παρέχει και αρκετά προηγμένα στοιχεία, όπως tabbed panels, scroll panes, trees, tables, lists. Επίσης παρέχει εμφανίσεις (look and feels) που μιμούνται την εμφάνιση, άλλων πλατφόρμων, όπως επίσης και look and feels τα οποία δεν παραπέμπουν σε κάποια ήδη υπάρχουσα εμφάνιση. Σε αντίθεση με τα αντικείμενα του AWT, τα Swing αντικείμενα δεν έχουν υλοποιηθεί για μια συγκεκριμένη πλατφόρμα. Αντ 'αυτού, είναι υλοποιημένα εξολοκλήρου σε Java και ως εκ τούτου είναι ανεξάρτητα από την πλατφόρμα. Στην παρακάτω εικόνα παρουσιάζεται μια διεπαφή η οποία είναι υλοποιημένη με Swing αντικείμενα. Η διεπαφή αποτελείται απο ένα πλήθος αντικειμένων όπως radioButtons, combobox, ένα απλό κουμπί, ένα πεδίο για την εισαγωγή κειμένου μεγάλου μήκους (TextArea) ένα πεδίο για εισαγωγή μικρού κειμένου (TextField) κι ένα πεδίο στο οποίο ο χρήστης εισάγει κάποιο κωδικό. Όλα τα αντικείμενα περιέχονται μέσα σε τρία αντικείμενα (Containers). Ένα Panel το οποίο περιέχει τα radiobuttons το κουμπί και τη λίστα, ένα TabbedDialogBox το οποίο έχει τα κουμπιά επιλογής (checkboxes) και το slider. Το τρίτο αντικείμενο είναι ένα δεύτερο panel το οποίο περιέχει τους δυο containers καθώς και τα υπόλοιπα αντικείμενα που αναφέρθηκαν προηγουμένως. Ακόμα στο πάνω μέρος του παραθύρου υπάρχει ένα menu με δυο επιλογές.



Εικόνα 3: Παράδειγμα διεπαφής σχεδιασμένη με το Swing

Εργαλειοθήκη SWT

Το SWT είναι ένα GUI toolkit widget. Αρχικά αναπτύχθηκε από την IBM και τώρα συντηρείται από το Eclipse Foundation σε συνεργασία με το Eclipse IDE. Είναι μια εναλλακτική λύση για το AWT και Swing Java εργαλείων για την δημιουργία GUIs. Το SWT είναι επίσης γραμμένο σε Java. Για την εμφάνιση GUI αντικειμένων, το SWT αποκτά πρόσβαση στις GUI βιβλιοθήκες του λειτουργικού συστήματος, χρησιμοποιώντας το JNI (Java Native Interface) με τρόπο που να είναι παρόμοιος με αυτόν που χρησιμοποιούν προγράμματα τα οποία είναι γραμμένα χρησιμοποιώντας τα APIs ενός συγκεκριμένου λειτουργικού συστήματος. Το SWT σχεδιάστηκε να είναι ένα "υψηλής απόδοσης" GUI toolkit, γρηγορότερο, με καλύτερη ανταπόκριση και να δεσμεύσει λιγότερους πόρους από ότι το Swing. Οι ρίζες του SWT βρίσκονται στην δουλεία της OTI η οποία ιδρύθηκε το 1988 και εξαγοράστηκε απο την IBM το 1996, η οποία δημιούργησε το SWT. Αντίθετα από τα AWT και Swing το SWT δεν είναι διαθέσιμο για κάθε πλατφόρμα η οποία υποστηρίζει Java αφού δεν είναι μέρος του Java release. Υπάρχουν επίσης ενδείξεις ότι το SWT σε πλατφόρμες εκτός των Windows είναι αισθητά λιγότερο αποτελεσματικό. Από όταν το SWT άρχισε να χρησιμοποιεί διαφορετικές βιβλιοθήκες για κάθε πλατφόρμα οι προγραμματιστές μπορεί να εκτεθούν σε bugs τα οποία οφείλονται στις περισσότερες (low level) λεπτομέρειες τις οποίες διαχειρίζονται απο ότι συμβαίνει στο Swing. Αυτό συμβαίνει επειδή το SWT είναι τεχνικά ένα στρώμα πάνω από τη βιβλιοθήκη που παρέχει τη GUI λειτουργικότητα, εκθέτοντας τους προγραμματιστές στη δημιουργία κώδικα σα μέρος της σχεδίασης μίας διεπαφής. Στην παρακάτω εικόνα φαίνεται μια SWT διεπαφή και η αναπαράστασή της σε διαφορετικά λειτουργικά συστήματα. Η διεπαφή περιέχει ένα tabbed pane με δύο καρτέλες μια με τίτλους (widget, toolkit), η καρτέλα περιέχει ένα πίνακα, ένα κουμπί επιλογής checkbox κι ένα κουμπί με κείμενο "Awesome", ενώ σα τίτλο το παράθυρο γράφει Vista.



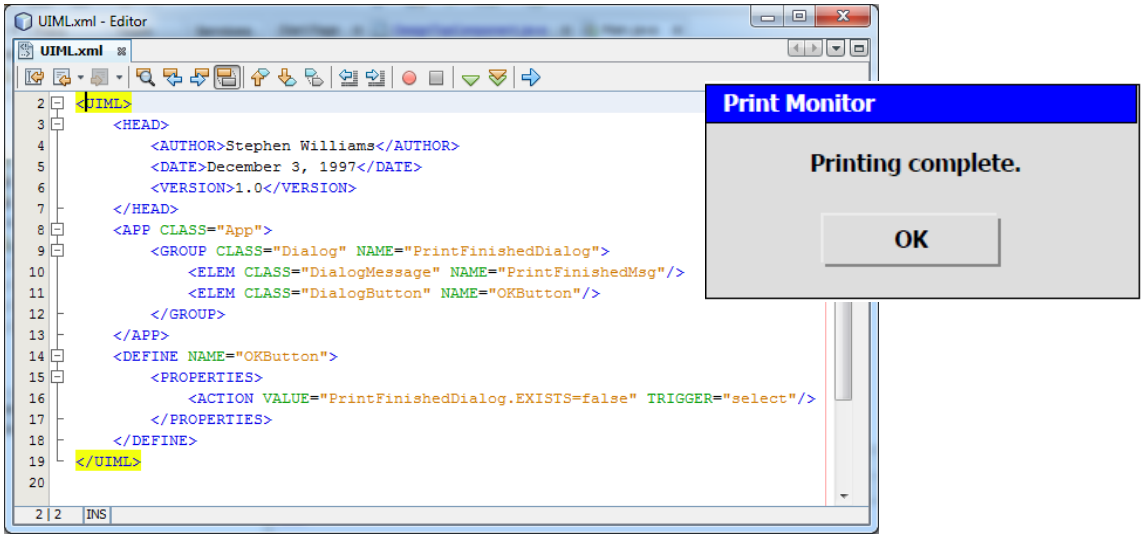
Εικόνα 4: Παράδειγμα διεπαφής σχεδιασμένη με το SWT

2.1.1 Μοντελοκεντρική Ανάπτυξη (Model Based UI Development)

Σκοπός του Model Based UI Development είναι να χωρίσει την διαδικασία ανάπτυξης διεπαφών σε επίπεδα αφαίρεσης, εστιάζοντας μόνο σε συγκεκριμένες λεπτομέρειες του κάθε επιπέδου ώστε η σχεδίαση να είναι απλούστερη επιτρέποντας στους σχεδιαστές να επικεντρωθούν στα σημαντικότερα μέρη της εφαρμογής χωρίς να αποσπούνται από σχεδιαστικές λεπτομέρειες. Έτσι με τη χρήση μοντέλων οι σχεδιαστές μπορούν να διαχειριστούν την αυξημένη πολυπλοκότητα τέτοιων εφαρμογών ευκολότερα κατά τη διάρκεια της υλοποίησης και να κάνουν ευκολότερα τροποποιήσεις.

UIML

Η UIML (User Interface Markup Language) είναι μια γλώσσα βασισμένη σε XML η οποία χρησιμοποιείται για την περιγραφή διεπαφών. Δημιουργήθηκε από την Virginia Tech το 1997. Στόχος της είναι να μειώσει την προσπάθεια που απαιτείται για την ανάπτυξη διεπαφών. Επιτρέπει την περιγραφή μιας διεπαφής σε δηλωτική όρους (δηλαδή ως κείμενο) και αφηρημένα. Αφηρημένα σημαίνει ότι δεν ορίζει πως ακριβώς θα φαίνονται οι διεπαφές αλλά ποία αντικείμενα θα εμφανίζονται και πως θα συμπεριφέρονται. Στην εικόνα που ακολουθεί φαίνεται ο κώδικας που απαιτείται για την εμφάνιση ενός παραθύρου, και πως αυτός μεταφράζεται σα διεπαφή.

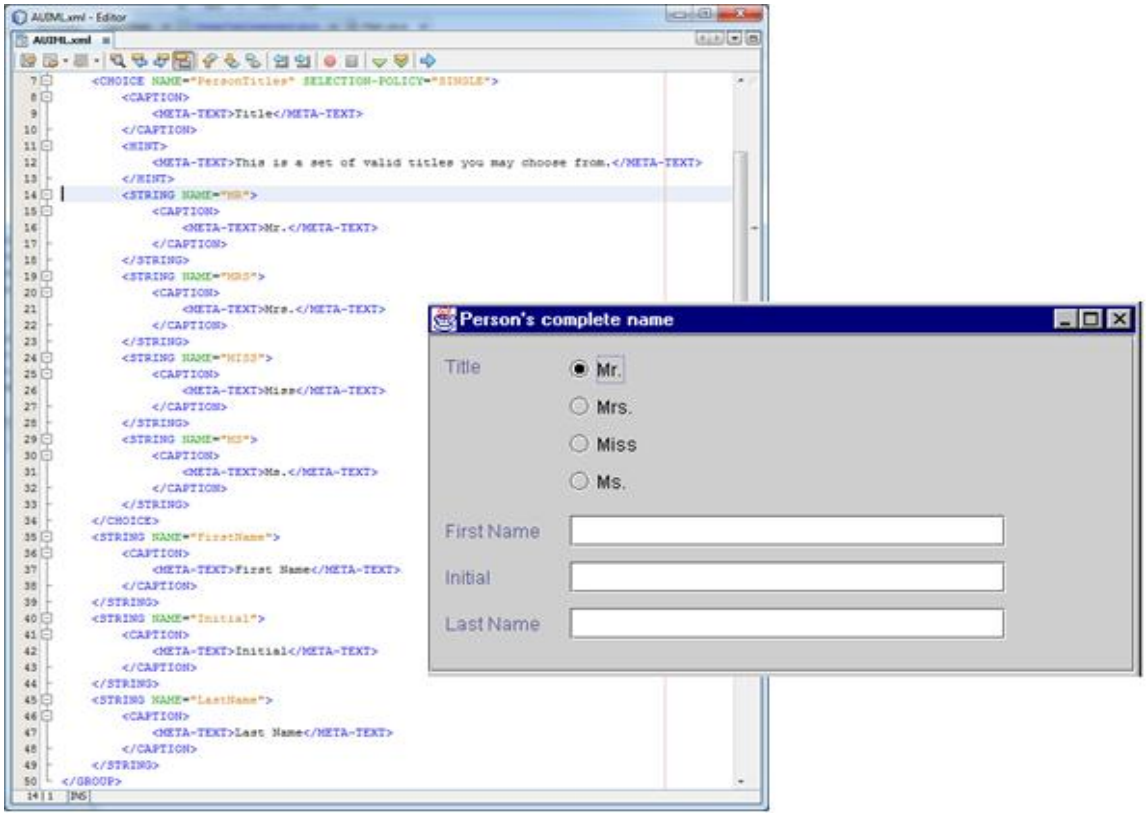


Εικόνα 5: UIML

Θεωρητικά θα μπορούσαμε να χρησιμοποιήσουμε αυτή τη περιγραφή για να δημιουργήσουμε διεπαφές για διαφορετικές πλατφόρμες όπως για PDAs. Στην πράξη όμως οι διαφορετικές δυνατότητες-απαιτήσεις που έχουν οι πλατφόρμες αυτές κάνουν τη συνολική μετατροπή δύσκολη. Άλλες λιγότερο φιλόδοξες γλώσσες προγραμματισμού προσπαθούν μόνο να περιγράψουν τις διεπαφές (ή άλλα μέρη μιας εφαρμογής) σε έναν ένα μόνο περιβάλλον, π.χ Windows.

AUIML

Η AUIML είναι μία γλώσσα περιγραφής διεπαφών βασισμένη στην XML. Είναι μια πλατφόρμα η οποία παρέχει μια τεχνολογικά ουδέτερη απεικόνιση των πάνελ, ιδιοτήτων, κλπ. Η AUIML καταγράφει πληροφορίες σχετικές τοποθέτηση των GUI αντικειμένων και αναθέτει την εμφάνιση τους στο renderer της πλατφόρμας. Ανάλογα με την πλατφόρμα ή τη συσκευή που χρησιμοποιείται, ο renderer αποφασίζει ποιός είναι ο καλύτερος τρόπος για να παρουσιάσει το GUI στο χρήστη και να λάβει είσοδο (αλληλεπίδραση) από το χρήστη. Η AUIML επιτρέπει στους προγραμματιστές να γράφουν μια εφαρμογή μια φορά και τρέχει στο Swing ή στο διαδίκτυο χωρίς καμία αλλαγή.



Εικόνα 6: AUIML

2.1.2 Μοντελοκεντρική Μηχανική Ανάπτυξης Διεπαφών (Model-based UI Engineering)

TeresaXML

Η TeresaXML είναι μια γλώσσα βασισμένη στην XML που αναπτύχθηκε στα πλαίσια του έργου Teresa, το οποίο υλοποιήθηκε από την HCI ομάδα του ISTI-CNR (http://giove.cnuce.cnr.it). Παρέχει ένα περιβάλλον που υποστηρίζει το σχεδιασμό και την παραγωγή ενός συγκεκριμένου τύπου διεπαφής για συγκεκριμένο τύπο πλατφόρμας. Η Teresa XML αποτελείται από δύο μέρη: (i) μια XML περιγραφή της σημειογραφίας CTT, η οποία ήταν η πρώτη γλώσσα XML για τα μοντέλα δραστηριοτήτων, (ii) μια γλώσσα για την περιγραφή διεπαφών. Για την περιγραφή των UI διευκρινίζει πώς τα διάφορα AIO (Abstract Interactive Objects) που συνθέτουν το UI είναι οργανωμένα, μαζί με την προδιαγραφή του διαλόγου UI. Πράγματι, ένα UI είναι ένα σύνολο από ένα ή περισσότερα στοιχεία παρουσίασης. Κάθε στοιχείο χαρακτηρίζεται από μια δομή, που περιγράφει την οργάνωση του UI και καμία ή περισσότερες συνδέσεις, πράγμα που δίνει πληροφορίες για τις σχέσεις μεταξύ των διαφόρων στοιχείων της διεπαφής. Κάθε στοιχείο μπορεί να είναι είτε ένα στοιχειώδες AIO ή μια σύνθεση του. Κάθε AIO μπορεί να είναι είτε μια αλληλεπίδραση AIO είτε μια AIO εφαρμογή ανάλογα με το αν υπάρχει ή όχι μια αλληλεπίδραση μεταξύ του χρήστη και της εφαρμογής. Η TeresaXML χρησιμοποιείται σε ένα tool (TERESSA), το οποίο υποστηρίζει τη δημιουργία task models, abstract UIs και running UIs. Το εργαλείο αυτό βρίσκεται ακόμα υπό κατασκευή, επομένως διεπαφές οι οποίες δημιουργούνται σε αυτό το εργαλείο μπορεί να περιέχουν λάθη.

Το Πλαίσιο Αναφοράς Χαμελέον (The Cameleon Reference Framework: 'CRF')

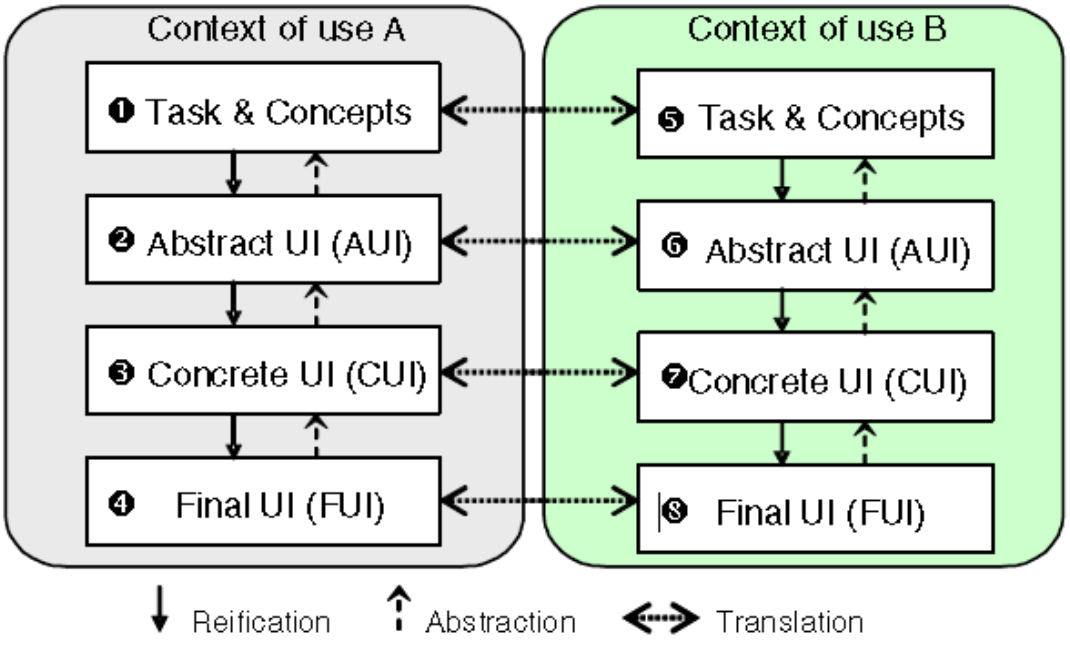
Όπως αναφέρθηκε προηγουμένως υπάρχει ένα σύνολο προβλημάτων και περιορισμών για την ανάπτυξη διεπαφών για πολλαπλά περιβάλλοντα. Μια προσέγγιση για τη λύση του συνόλου των προβλημάτων αυτών περιγράφεται απο το Cameleon Reference Framework. Η λογική του βασίζεται στο χωρισμό του κύκλου σχεδιασμού μιας διεπαφής σε στάδια. Σύμφωνα με το CAMELEON (**C**ontext **A**ware **M**odelling for **E**nabling and **L**everaging **E**ffective interacti**ON**) ο κύκλος σχεδιασμού χωρίζεται σε τέσσερα επίπεδα αφαίρεσης (Task & Concepts, AUI, CUI, FUI) έτσι ώστε να απλοποιηθούν και να απομονωθούν τα εμπόδια που συναντώνται κατά τη σχεδίαση και να αντιμετωπιστούν ευκολότερα. Τα τέσσερα επίπεδα αφαίρεσης για το κύκλο σχεδιασμού μιας εφαρμογής:

i. **Μοντέλο διεργασίας και αντικειμένων** (Task and Object model) **:** Σε αυτό το επίπεδο περιγράφονται οι λογικές δραστηριότητες που πρέπει να εκτελέσουν οι χρήστες για να επιτύχουν τους στόχους τους. Οι εργασίες ενίοτε περιγράφονται ιεραρχικά και καθορίζουν με ποια σειρά θα πρέπει ο χρήστης να εκτελέσει τις διεργασίες για να επιτύχει τους στόχους του.

ii. **Μοντέλο αφηρημένης διεπαφής** (Abstract User Interface): Σε αυτό το επίπεδο καθορίζεται η λογική δομή της διεπαφής.

iii. **Μοντέλο συγκεκριμένης γραφικής διεπαφής** (Concrete User Interface): Σε αυτό το σημείο κάθε αφηρημένη διεπαφή αντικαθίσταται με μια συγκεκριμένη γραφική διεπαφή, η οποία εξαρτάται από τον τύπο της πλατφόρμας και έχει συγκεκριμένες ιδιότητες η οποίες προσδιορίζουν πώς θα πρέπει να παρουσιαστεί στο χρήστη.

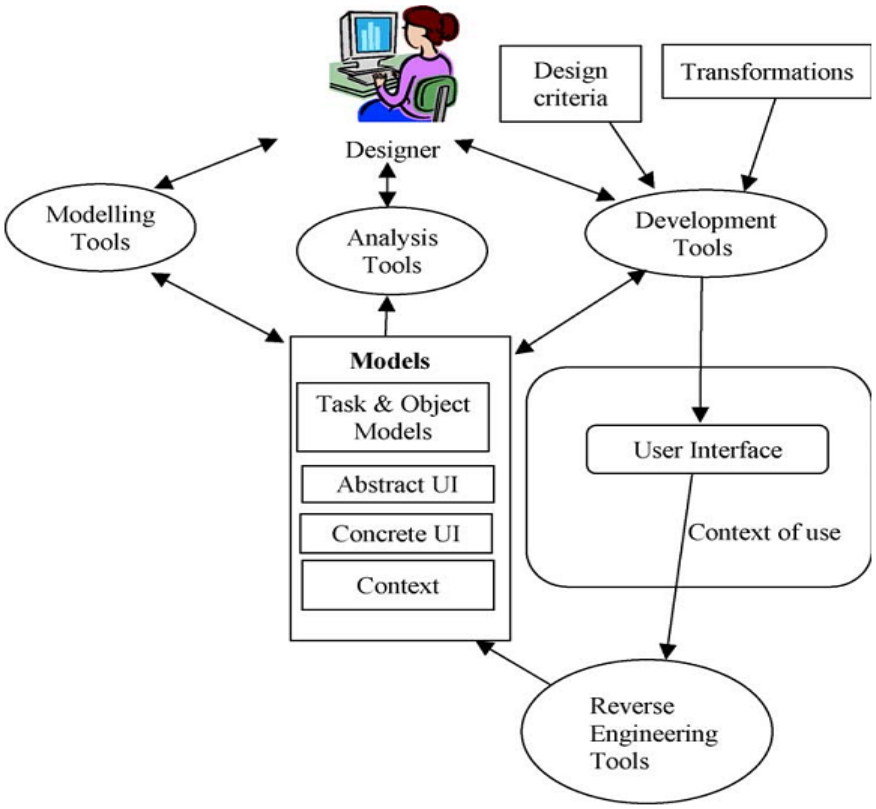
iv. **Τελική γραφική Διεπαφή** (Final User Interface): Στο τελευταίο επίπεδο αφαίρεσης η συγκεκριμένη διεπαφή, μεταφράζεται σε μια γραφική η οποία ορίζεται από μια συγκεκριμένη γλώσσα προγραμματισμού (π.χ. XHTML, Java).



Εικόνα 7: Επίπεδα αφαίρεσης

Για την καλύτερη κατανόηση αυτών των επιπέδων αφαίρεσης μπορούμε να θεωρήσουμε ως παράδειγμα μιας εργασίας: την δημιουργία κράτησης σε ένα ξενοδοχείο. Αυτό το έργο μπορεί να αναλυθεί σε επιλογή ημερομηνίας άφιξης και αναχώρησης και άλλες δευτερεύουσες εργασίες. Στο αφηρημένο επίπεδο διεπαφής χρήστη θα πρέπει να προσδιορίσει το διαδραστικό αντικείμενο που απαιτείται για την υποστήριξη της διεργασίας. Όταν προχωράμε στο επίπεδο συγκεκριμένης γραφικής διεπαφής, πρέπει να εξετάσουμε τα συγκεκριμένο αντικείμενα αλληλεπίδρασης τα οποία υποστηρίζονται. Έτσι, σε μια desktop εφαρμογή, η επιλογή μπορεί να υποστηριχθεί από ένα γραφικό αντικείμενο όπως μια λίστα ή ένα comboBox. Αυτή η επιλογή είναι πιο αποτελεσματική από άλλες, επειδή τα αντικείμενα αυτά δίνουν στο χρήστη τη δυνατότητα να επιλέξει μία επιλογή, στο συγκεκριμένο παράδειγμα μια ημερομηνία, από ένα πλήθος επιλογών. Η τελική διεπαφή χρήστη είναι το αποτέλεσμα τέτοιων επιλογών καθώς και άλλων όπως ο τύπος και το μέγεθος της γραμματοσειράς, τα χρώματα ή οι διακοσμητικές εικόνες. Για παράδειγμα, μπορεί να εμφανιστεί η λίστα με τη μορφή ενός ημερολογίου.

Πολλές μετατροπές είναι δυνατόν να πραγματοποιηθούν μεταξύ αυτών τεσσάρων επίπεδων για κάθε πλατφόρμα αλληλεπίδρασης. Από το υψηλότερο επίπεδο αφαίρεσης σε πιο συγκεκριμένο ή αντίστροφα ή ακόμα και μεταξύ το ίδιων επίπεδων αφαίρεσης, αλλά για διαφορετικούς τύπους πλατφόρμας ή ακόμα και με οιοδήποτε συνδυασμό αυτών. Κατά συνέπεια, μια ευρεία ποικιλία των καταστάσεων μπορεί να αντιμετωπιστεί. Γενικότερα, η δυνατότητα σύνδεσης των πτυχών που σχετίζονται με τα στοιχεία διεπαφής χρήστη σε περισσότερο σημασιολογικές πτυχές, δίνοντας τη δυνατότητα στα έξυπνα εργαλεία να βοηθήσουν στο σχεδιασμό, την αξιολόγηση, και τον χρόνο εκτέλεσης. Στην Εικόνα 8 παρουσιάζεται η αξιοποίηση των μοντέλων κατά τη φάση της σχεδίασης.



Εικόνα 8: Παράδειγμα κράτησης ξενοδοχείου

UsiXML

Η UsiXML (USer Interface eXtensible Markup Language) είναι μια γλώσσα περιγραφής διεπαφών που βασίζει το συντακτικό της στην XML. Είναι δομημένη σύμφωνα με τα τέσσερα διαφορετικά επίπεδα αφαίρεσης όπως ορίζονται από το Cameleon Reference Framework όντας το 'Reference Implementation' αυτού. Ως εκ τούτου η UsiXML δίνει τη δυνατότητα περιγραφής διεπαφών που είναι είτε 'Computation Independent', είτε Platform & Modality independent, είτε 'Platform Specific'. Ένα άλλο χαρακτηριστικό της UsiXML είναι η υποστήριξη πολυκαναλικών διεπαφών, δηλαδή διεπαφών που χρησιμοποιούν - εκμεταλλεύονται ταυτόχρονα πλήθος εναλλακτικών καναλιών αλληλεπίδρασης (γραφικό, ακουστικό, κοκ.). Ένα άλλο σημαντικό χαρακτηριστικό της UsiXML είναι η υποστήριξη πλαστικών διεπαφών δηλαδή διεπαφών που μπορεί να εξελιχθούν ανάλογα με το τρέχων περιβάλλον χρήσης έχοντας ως γνώμονα τη διατήρηση της ευχρηστίας και της συνέχειας στη χρήση.

Η UsiXML τέλος βασίζεται στην εφαρμογή έτοιμων –προκαθορισμένων, μετασχηματισμών ώστε να είναι δυνατή η αυτοματοποιημένη μετάβαση μεταξύ των διαφορετικών επιπέδων αφαίρεσης κατά τη σχεδίαση.

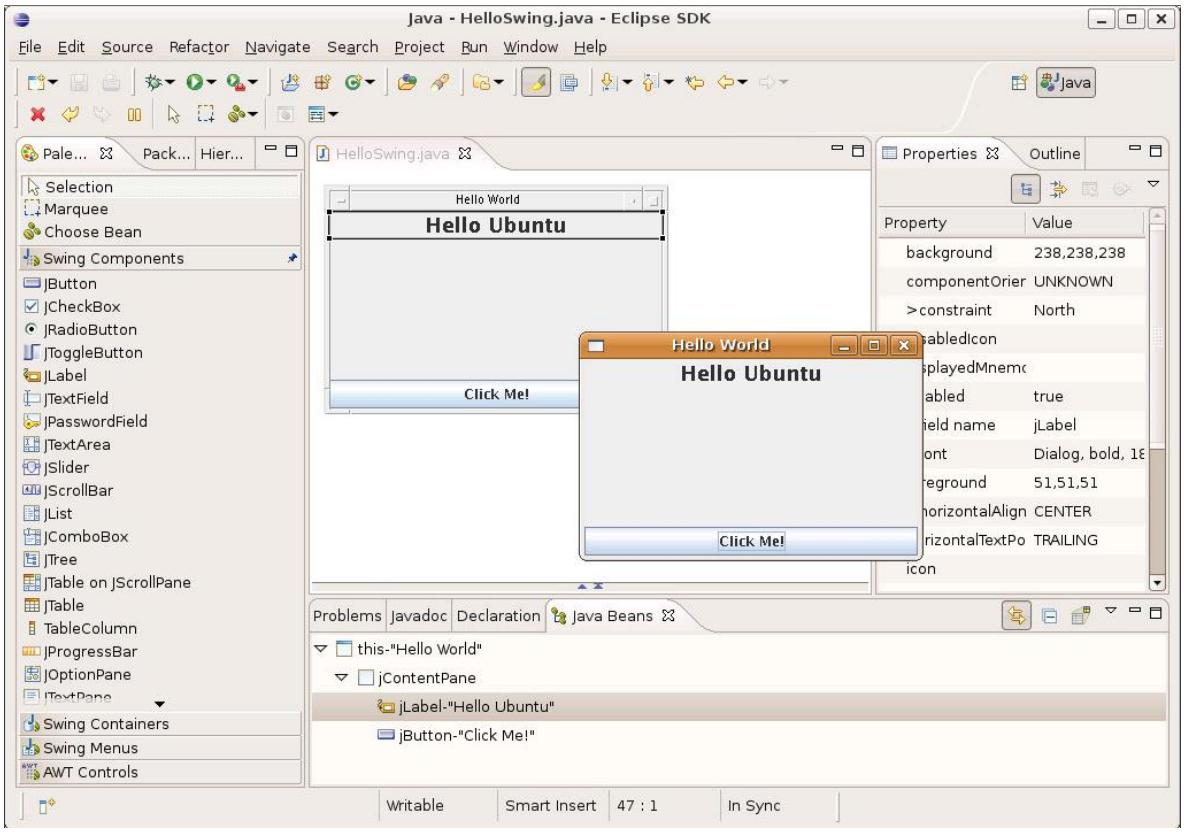
MariaXML

Η MariaXML αποτελεί υλοποίηση του Cameleon Reference Framework και αντικαταστάτη της TeresaXML έχοντας ως κύριο στόχο την παροχή δυνατότητας γηγενούς υποστήριξης περιγραφής και ενσωμάτωσης Service Oriented Architectures (SOAs). Παράλληλα φτιάχτηκε ώστε να υποστηρίζει δυναμικές συμπεριφορές, events, και εμπλουτισμένες διαδικτυακές εφαρμογές, δίνοντας τη δυνατότητα διεπαφές η οποίες περιγράφονται σε MariaXML να μπορούν να ενσωματωθούν σε διαδικτυακές υπηρεσίες.

3. Εργαλεία Υποστήριξης Φάσης Σχεδίασης

Eclipse IDE: Eclipse Window Builder

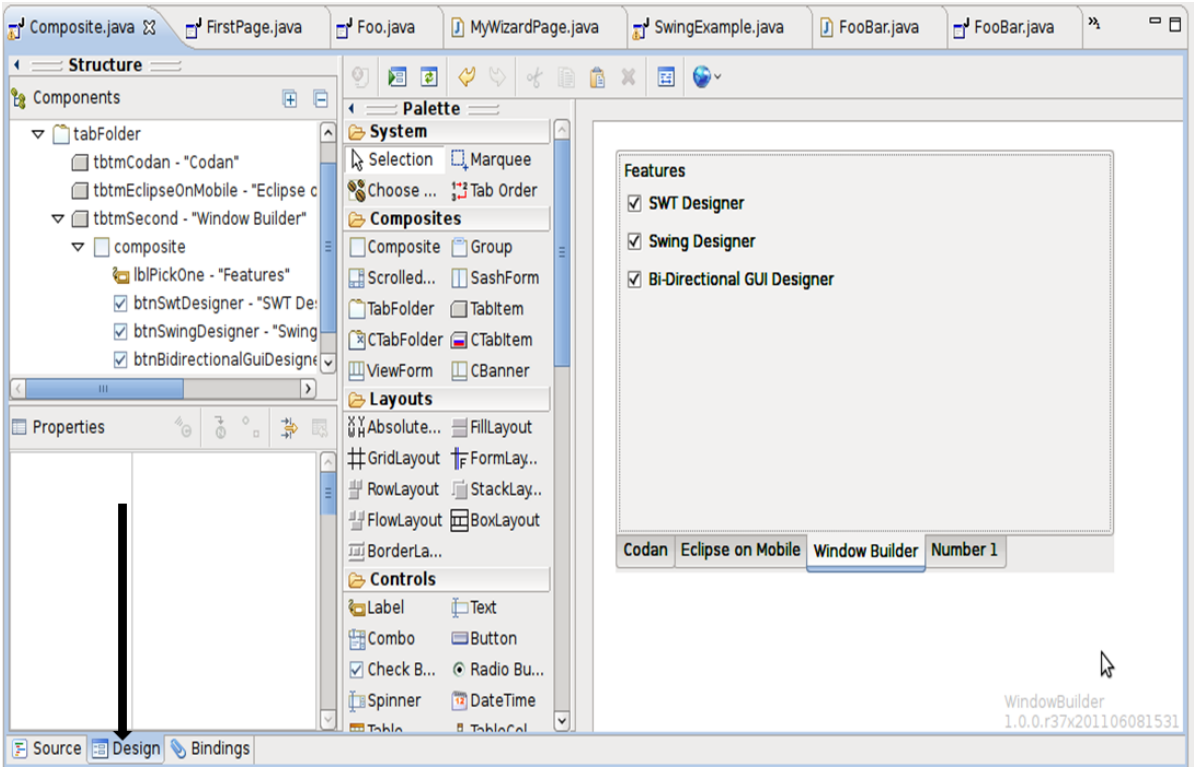
Το Eclipse (Εικόνα 9) είναι ένα πολυγλωσσικό περιβάλλον ανάπτυξης λογισμικού που αποτελείται από ένα ολοκληρωμένο περιβάλλον ανάπτυξης (IDE) και ένα επεκτάσιμο plug-in σύστημα. Είναι γραμμένο κυρίως σε Java και μπορεί να χρησιμοποιηθεί για την ανάπτυξη εφαρμογών σε Java και, μέσω των διαφόρων plug-ins, άλλων γλωσσών προγραμματισμού όπως Ada, C, C + +, PHP, Python, Groovy και Scheme. Οι χρήστες μπορούν να επεκτείνουν τις ικανότητές του IDE αυτού με την εγκατάσταση ήδη έτοιμων plug-ins που γράφονται για το Eclipse, όπως το kit εργαλείων ανάπτυξης και για άλλες γλώσσες προγραμματισμού, η μπορούν να δημιουργήσουν τα δικά τους plug-in. Είναι δωρεάν λογισμικό ανοιχτού κώδικα και ένα από τα πρώτα IDE για να τρέχει σε GNU classpath. Το eclipse εμφανίστηκε αρχικά ως πρότζεκτ της IBM, αναπτύχθηκε αρχικά από την Object Technology International (OTI) σαν ο Java-based αντικαταστάτης της VisualAge οικογένειας IDE που βασίζονταν στη Smalltalk, η οποία επίσης είχε αναπτυχθεί από την OTI. Το Eclipse μέσω της υποστήριξης δημιουργίας Java εφαρμογών δίνει τη δυνατότητα στους χρήστες να σχεδιάσουν διεπαφές οι οποίες είναι ανεξάρτητες του λειτουργικού στο οποίο τρέχουν όμως δεν προσφέρει κάποιο δικό του GUI Builder, ωστόσο υπάρχουν διάφορα δωρεάν εργαλεία για την υποστήριξη της σχεδίασης διεπαφών.



Εικόνα 9: Περιβάλλον χρήσης Eclipse

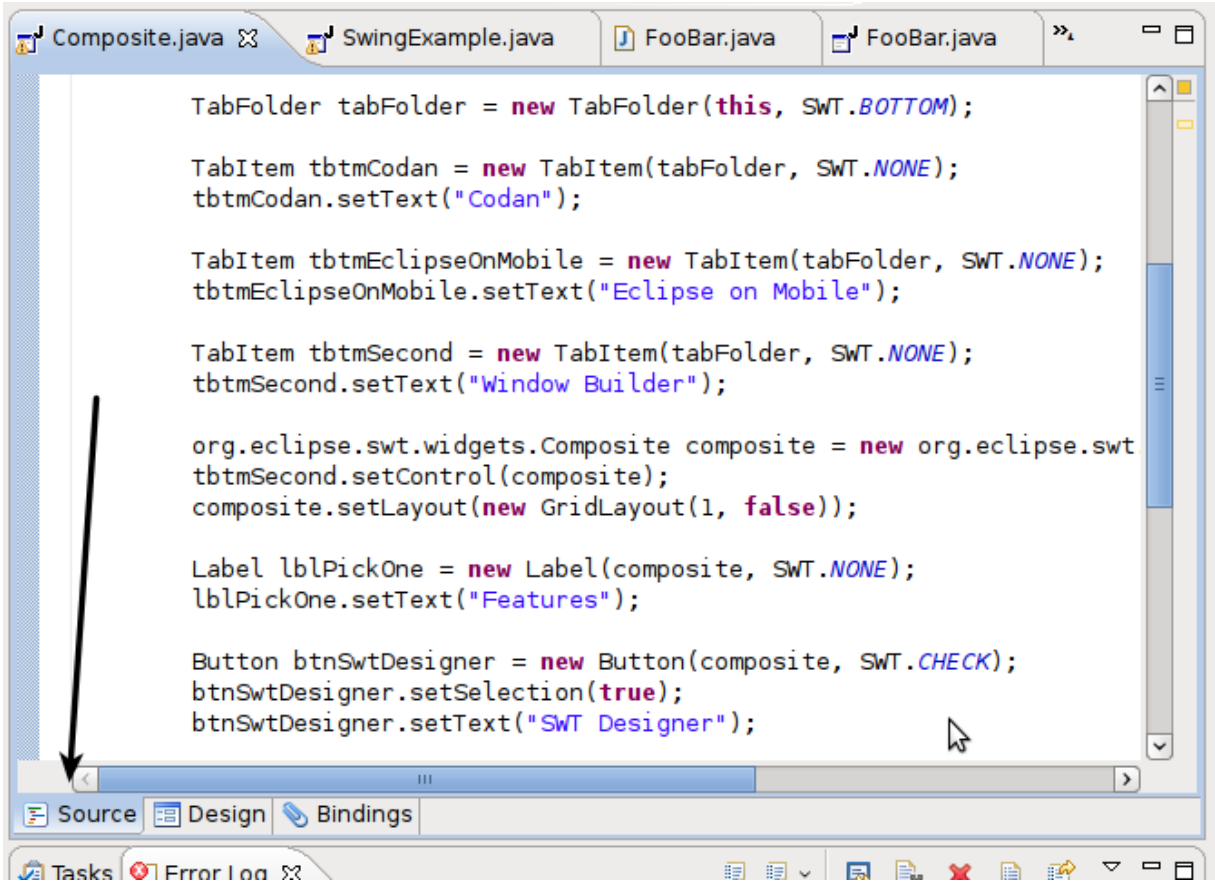
Ένα τέτοιο εργαλείο είναι το WindowBuilder το οποίο υποστηρίζει τα Eclipse SWT, Sun Swing και GWT της Google για τη σχεδίαση διεπαφών με πολύ γρήγορο κι απλό τρόπο χωρίς οι χρήστες να χρειάζεται να αφιερώσουν πολύ χρόνο στο να γράφουν κώδικα. Αποτελείται απο δύο κύριες όψεις η οποίες είναι συγχρονισμένες μεταξύ τους. Η μία είναι η όψη όπου γίνεται η σχεδίαση η οποία προσφέρει έτοιμες λειτουργίες για την διαχείριση των διαχειριστών διάταξης και των διαχειριστών γεγονότων (event handlers), property editor για πιο εύκολη αλλαγή των ιδιοτήτων των αντικειμένων που χρησιμοποιούμε. Συνοδεύεται απο μια παλέτα που περιέχει τα αντικείμενα που μπορεί να χρησιμοποιήσει ο χρήστης για τη σχεδίαση μίας διεπαφής τα οποία τα χρησιμοποιεί κάνοντας drag & drop απο τη παλέτα στη σκηνή που γίνεται η σύνθεση της διεπαφής.

Στην Εικόνα 10 φαίνεται ένα παράδειγμα διεπαφής σχεδιασμένο στο Builder του Eclipse η οποία αποτελείται απο τέσσερις καρτέλες οι οποίες έχουν τίτλο (Codan, Eclipse on Mobile, Window Builder, Number 1). Στην εικόνα φαίνονται τα περιεχόμενα της τρίτης καρτέλας τα οποία είναι τρία CheckBoxes.



Εικόνα 10: Eclipse GUI Builder

Στην δεύτερη όψη Εικόνα 11 ο Builder δημιουργεί αυτόματα τον Java κώδικα που θα έπρεπε να γράψει ένας σχεδιαστής για να δημιουργήσει την διεπαφή. Εκεί αφού κάποιος έχει σχεδιάσει τη διεπαφή μπορεί να προσθέσει των κώδικα που χρειάζεται για την λειτουργία των αντικειμένων που έχει χρησιμοποιήσει. Η Εικόνα 11 παρουσιάζει το κώδικα που έχει δημιουργηθεί για την παραπάνω διεπαφή.



Εικόνα 11: Παραγόμενος κώδικας Eclipse Builder

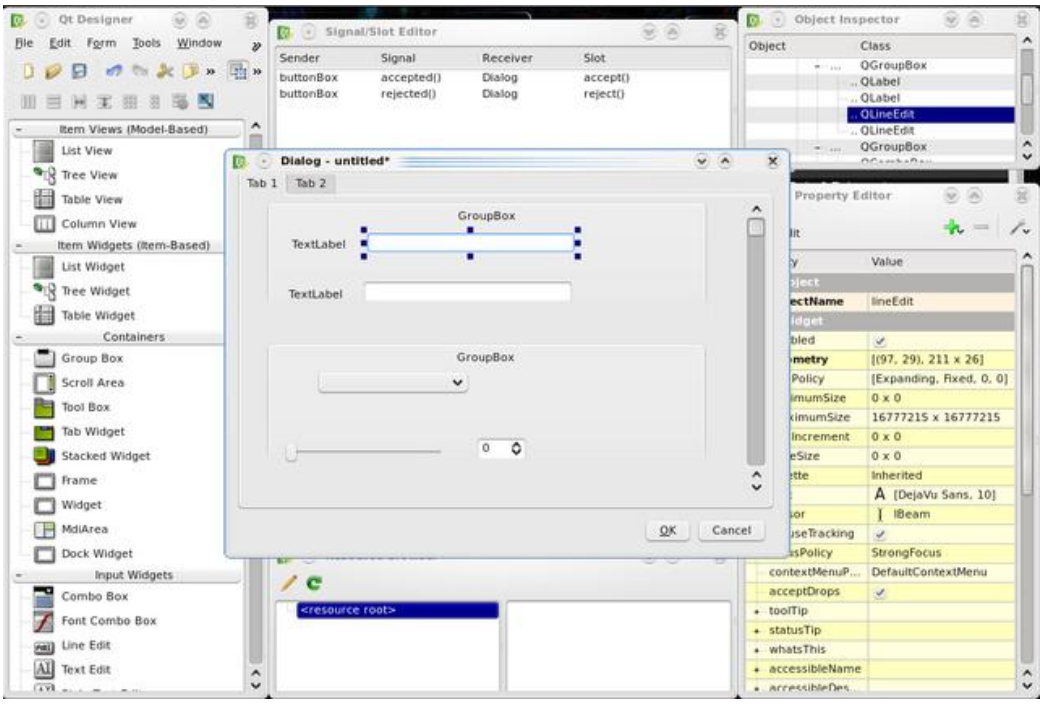
Qt Creator IDE: Qt Designer

Το ολοκληρωμένο περιβάλλον ανάπτυξης Qt (Εικόνα 12) είναι ένα IDE για συμβατό με όλες τις πλατφόρμες, ενώ αποτελεί μέρος του Qt SDK. Υποστηρίζει την C++ και την QML γλώσσα προγραμματισμού. Το Qt υποστηρίζει τη δημιουργία εφαρμογών για Desktop περιβάλλοντα (Windows, Linux, FreeBSD and Mac OS) όπως και για mobile (Symbian,Maemo, και MeeGo). Κατασκευάστηκε από την Nokia και χρησιμοποιείται από εταιρίες όπως Adobe Photoshop Elements, Skype, VLC media player, European Space Agency, Google, HP, Panasonic, Samsung, Volvo. Είναι λογισμικό ανοιχτού κώδικα και όλες οι εκδόσεις του υποστηρίζουν ένα ευρύ φάσμα compilers, συμπεριλαμβανομένου του GCC,C ++ compiler και το Visual Studio suite. Παρέχει λειτουργίες στις οποίες περιλαμβάνονται η πρόσβαση σε βάσεις δεδομένων SQL, το XML parsing, thread management, network support, όπως κι και ένα ενιαίο cross-platform API για τη διαχείριση των αρχείων. Για το σχεδιασμό διεπαφών προσφέρει δυο editors τον Qt Designer και τον Qt Quick Designer.

Ο Qt Designer είναι ένα εργαλείο το οποίο βοηθάει το χρήστη να σχεδιάζει διεπαφές χρησιμοποιώντας τα Qt widgets, απλά δουλεύοντας με drag & drop. Τα widgets της βασικής βιβλιοθήκης μπορούν να προσαρμοστούν ανάλογα με τις ανάγκες του χρήστη (αλλαγή style, ή ανάλυσης). Τέλος ο Qt Designer είναι συμβατός και με το Visual Studio όπως και με το Eclipse.

Ο Qt Quick Designer χρησιμοποιείται για την δημιουργία εφαρμογών κι διεπαφών χρησιμοποιώντας την QML (Javascript) γλώσσα προγραμματισμού. Στην QML κάθε διεπαφή περιγράφεται σαν ένα δέντρο από

αντικείμενα τα οποία περιέχουν κάποιες ιδιότητες. Ο Qt Quick Designer παρέχει δύο όψεις στο χρήστη, μια για τη σχεδίαση της διεπαφής και μια στην οποία υπάρχει ο απαραίτητος κώδικας (QML) ο οποίος δημιουργείται κατά τη σχεδίαση των διεπαφών.



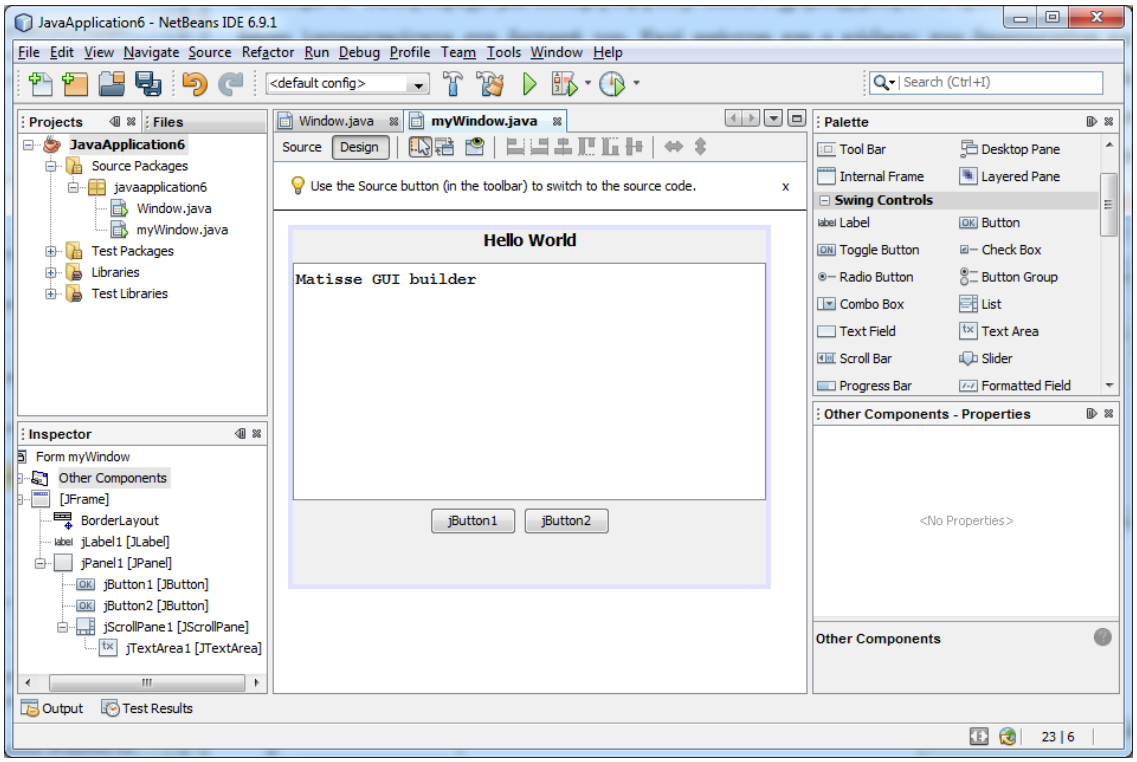
Εικόνα 12: Σχεδίαση διεπαφής με χρήση του Qt Designer

NetBeans IDE: Matisse GUI Builder

Το NetBeans είναι ένα ολοκληρωμένο σύστημα ανάπτυξης εφαρμογών ανοικτού κώδικα. Ξεκίνησε το 1996 ως Xelfi και ήταν ένα μαθητικό πρόγραμμα υπό την επίβλεψη του Faculty of Mathemetics and Physics στο Charles University στην Πράγα. Το 1997 ο Roman Staněk δημιούργησε μια εταιρία η οποία εμπορεύονταν το Netbeans IDE η οποία μετέπειτα αγοράσθηκε από την Sun Microsystems το 1999. Η Sun μετέτρεψε το Netbeans σε λογισμικό ανοικτού κώδικα τον Ιούνιο του 2000 κι από τότε συνεχώς επεκτείνεται. Το 2010 η Oracle εξαγόρασε τη Sun και κατ'επέκταση το Netbeans. Είναι γραμμένο σε Java και είναι συμβατό με όλες τις πλατφόρμες (Windows, Linux, Mac Os) στις οποίες έχει εγκατασταθεί JVM. Υποστηρίζει διάφορες γλώσσες προγραμματισμού όπως Java, C/C++, php, Javascript κ.α. Για την ανάπτυξη Java εφαρμογών απαιτείται ένα JDK. Ισχυρό πλεονέκτημα του NetBeans IDE είναι το γεγονός οτι μια εφαρμογή που χτίζεται σε αυτό, μπορεί να διαχωριστεί σε ενότητες (modules) με συσχετιζόμενες βιβλιοθήκες. Για την ανάπτυξη ενός ολοκληρωμένου εργαλείου βασισμένο στη συγκεκριμένη πλατφόρμα συνήθως χρησιμοποιείται η έννοια της σουίτας. Ουσιαστικά πρόκειται για την ομαδοποίηση και οργάνωση πολλών διαφορετικών modules τα οποία είναι εξαρτημένα μεταξύ τους. Για την εύκολη σχεδίαση διεπαφών το NetBeans παρέχει στους χρήστες του το Matisse GUI Builder (Εικόνα 13), το οποίο βοηθάει στο σχεδιασμό διεπαφών με γραφικό τρόπο.

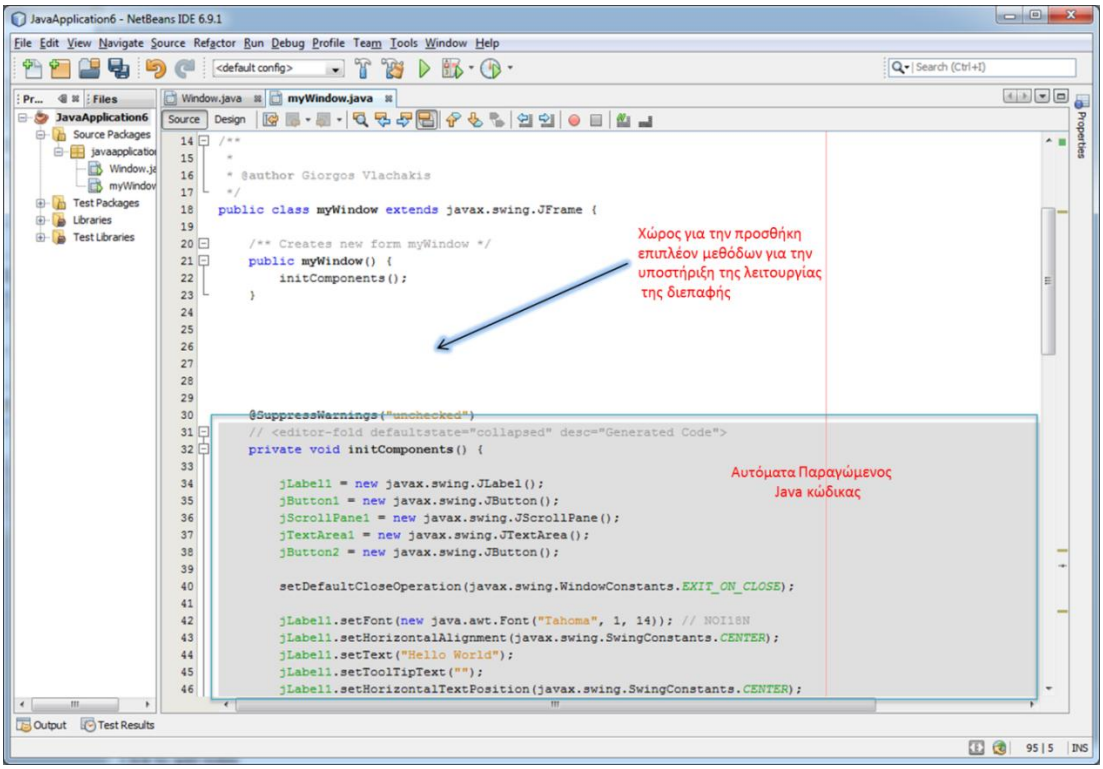
Το Matisse χρησιμοποιεί την παλέτα του NetBeans στην οποία βρίσκονται τα διαθέσιμα αντικείμενα. Επιπλέον προσφέρει έτοιμες λειτουργικότητες όπως η σύνδεση ενός αντικειμένου με ένα

listener, για την διαχείριση των γεγονότων. Καθώς ο χρήστης σχεδιάζει μία διεπαφή το Matisse του παρέχει οπτικές οδηγίες η οποίες αναφέρονται στη βέλτιστη απόσταση ή την ευθυγράμμιση των αντικειμένων που τοποθετεί ο χρήστης στη διεπαφή. Εκτός από τους συνηθισμένους διαχειριστές διάταξης το Matisse δίνει τη δυνατότητα στο χρήστη να χρησιμοποιήσει το "Free Design" το οποίο χρησιμοποιώντας το GroupLayout τοποθετεί τα αντικείμενα στη διεπαφή όπως ορίζει ο χρήστης και ταυτόχρονα ορίζει και τις συμπεριφορές τους. Έτσι καταφέρνει να λύσει ένα σημαντικό πρόβλημα των προγραμματιστών κατά τη δημιουργία GUI εφαρμογών, με τον εξορθολογισμό της ροής εργασίας, ελευθερώνοντας τους προγραμματιστές από την πολυπλοκότητα των διαχειριστών διάταξης (Layout managers) του Swing. Το Matisse βρίσκεται στη βασική έκδοση του NetBeans και προσφέρει τη δυνατότητα στους χρήστες εκτός απο τα βασικά αντικείμενα που χρησιμοποιεί, να χρησιμοποιήσουν αντικείμενα τα οποία έχουν υλοποιήσει οι ίδιοι ή αντικείμενα τα οποία προσφέρονται απο διάφορους άλλους χρήστες, δηλαδή οποιδήποτε αντικείμενο υλοποιήσουμε μπορούμε μέσω του palette manager να το προσθέσουμε στη παλέτα με τα υπόλοιπα αντικείμενα και να το χρησιμοποιούμε κατά τη σχεδίαση μιας διεπαφής. Στην Εικόνα 13 φαίνεται η σχεδίαση μιας διεπαφής με τη χρήση του Matisse.



Εικόνα 13: Σχεδίαση με το Matisse GUI builder

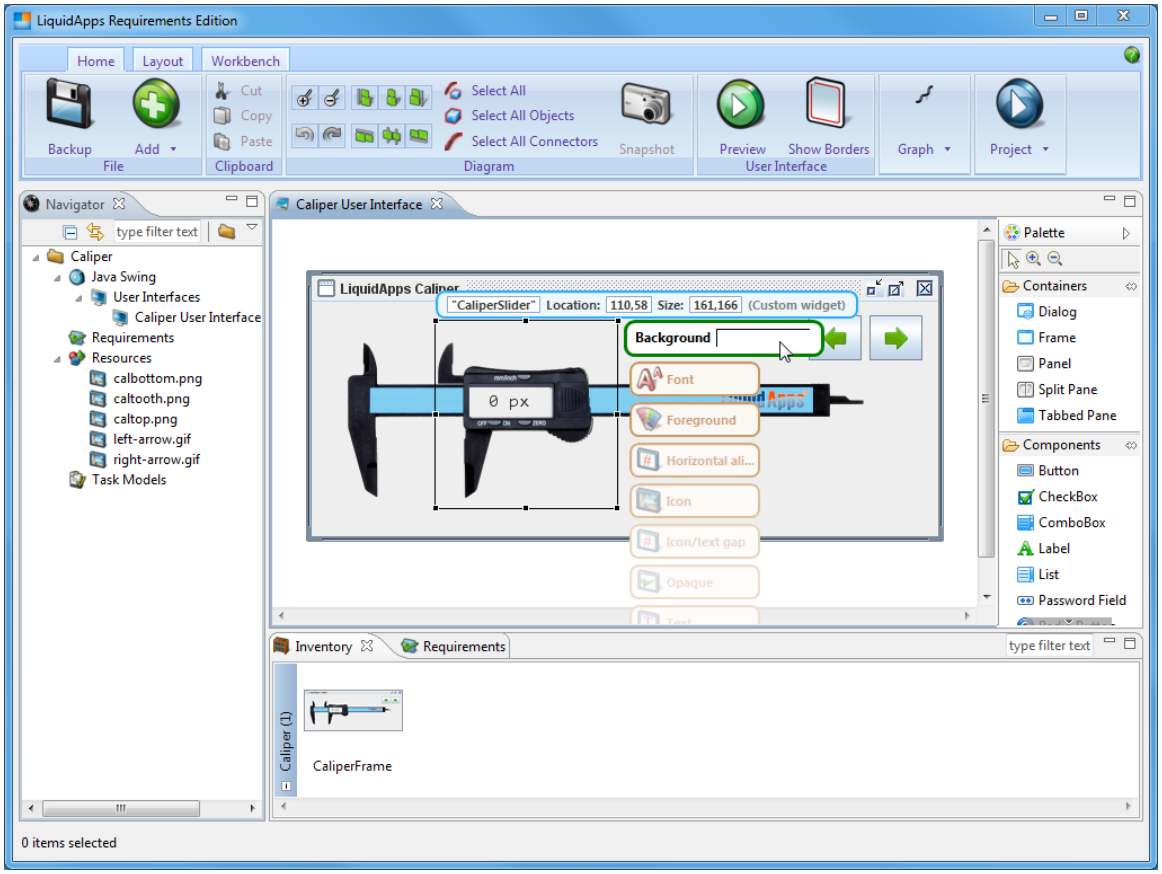
Εκτός απο την γραφική όψη το Matisse παρέχει και μια δεύτερη όψη Εικόνα 14 στην οποία ο χρήστης υλοποιεί τον Java κώδικα ο οποίος θα χρησιμοποιείται για τις λειτουργίες της διεπαφή. Στην όψη αυτή ο κώδικας χωρίζεται σε δυο μέρη, ένα είναι το μέρος το οποίο περιέχει τον αυτόματα υλοποιημένο κώδικα (γαλάζιο πλαίσιο) ο οποίος περιέχει πληροφορίες σχετικά με τα ονόματα των αντικειμένων, τη θέση τους στη διεπαφή κα. Κι σένα δεύτερο το οποίο περιέχει τις μεθόδους που δημιουργεί ο χρήστης. Στο σημείο του αυτόματα παραγόμενου κώδικα ο χρήστης δεν μπορεί να κάνει αλλαγές προγραμματιστικά παρά μόνο τροποποιώντας την διεπαφή (γραφικά).



Εικόνα 14: Java κώδικας που δημιουργείται κατά τη σχεδίαση

LiquidApps IDE

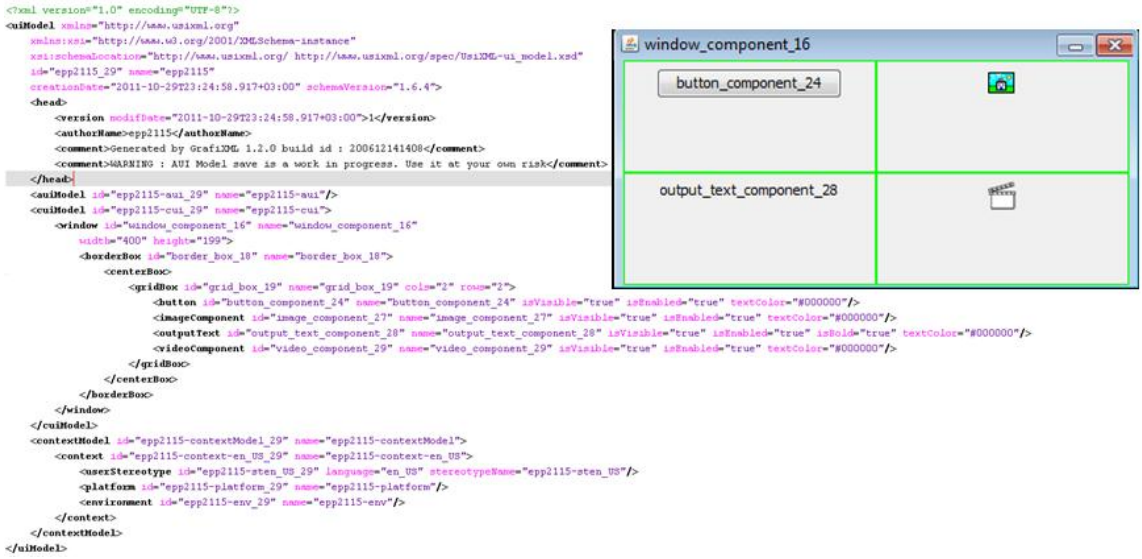
Το LiquidApps (Εικόνα 15), συνδυάζει το σχεδιασμό των UIs, την συγκέντρωση απαιτήσεων και την μοντελοποίηση καθηκόντων σε ένα εργαλείο. Επιτρέπει τη συνεργασία των σχεδιαστών, των προγραμματιστών, και των λοιπών ενδιαφερόμενων με σκοπό να παράγουν εφαρμογές γρήγορα. Το LiquidApps μπορεί να βοηθήσει τις ομάδες να αναπτύξουν νέο λογισμικό γρήγορα. Αλλά μπορεί επίσης να βοηθήσει με τη «μετανάστευση» του υπάρχοντος λογισμικού. Την μεταγλώττιση δηλαδή εφαρμογών από γλώσσες που τείνουν να εγκαταλειφθούν όπως η Ada σε πιο σύγχρονες πλατφόρμες. Επίσης διεπαφές που σχεδιάζονται στο LiquidApps μπορούν να εξαχθούν ως πακέτα τα οποία μπορούν να χρησιμοποιηθούν στο Eclipse η στο NetBeans.



Εικόνα 15: Περιβάλλον χρήσης LiquidApps

GrafiXML

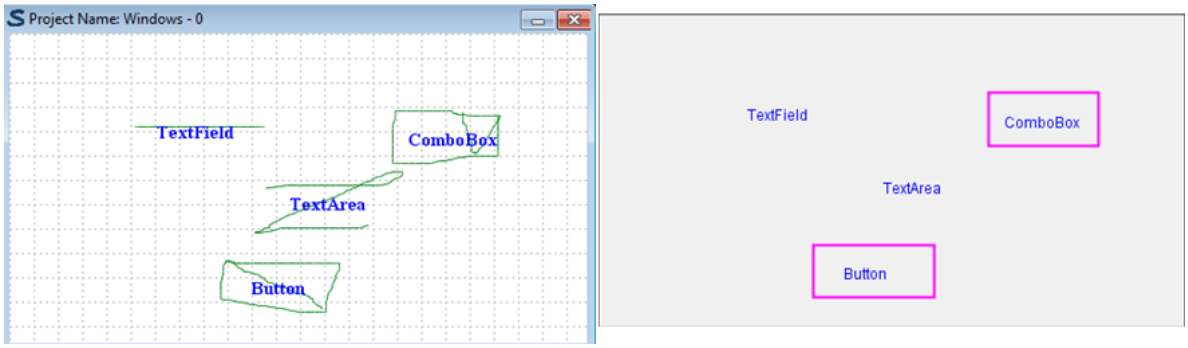
Το GrafiXML (Εικόνα 16) είναι ένα εργαλείο το οποίο αναπτύχθηκε απο την ομάδα της UsiXML και επιτρέπει στους χρήστες να σχεδιάζουν UIs για πολλαπλές πλατφόρμες. Οι χρήστες μπορούν να αποθηκεύσουν μία διεπαφή σε διάφορες μορφές όπως Java ή XHTML, συνηθέστερος όμως τρόπος αποθήκευσης είναι η αποθήκευση της διεπαφής σε UsiXML. Λειτουργεί όπως κάθε άλλη εφαρμογή σχεδίασης διεπαφών με την διαφορά ότι μεταχειρίζεται περισσότερες ιδιότητες widget, από τις υπόλοιπες εφαρμογές και ότι αποθηκεύει κάθε διεπαφή σε UsiXML αντί σε μια συγκεκριμένη μορφή κώδικα. Με αυτό το τρόπο, είναι δυνατή η διατήρηση πολλών διαφορετικών εκδόσεων μιας διεπαφής. Στα μειονεκτήματα της GrafiXML όμως είναι ότι δεν υλοποιεί όλα τα διαθέσιμα αντικείμενα της UsiXML.



Εικόνα 16: GrafiXML.

SketchiXML

Το SketchiXML είναι μια διαδραστική εφαρμογή η οποία αναπτύχθηκε απο την ομάδα της UsiXML και επιτρέπει στους σχεδιαστές και στους τελικούς χρήστες σχεδιάσουν-ζωγραφίσουν διεπαφές με διαφορετικά επίπεδα λεπτομέρειας. Η συμπεριφορά της εφαρμογής μπορεί να οριστεί με μια σειρά από παραμέτρους. Κάθε σχήμα που ζωγραφίζει ο χρήστης αναγνωρίζεται σαν ένα αντικείμενο το οποίο θα προστεθεί στη διεπαφή, η συσχέτιση των σκίτσων με τα αντικείμενα μπορεί να αλλάξει αφού ο χρήστης μπορεί να επιλέξει το ποιό αντικείμενο θα αντιπροσωπεύει το κάθε σχήμα που ζωγραφίζει. Όταν ο σχεδιασμός έχει ολοκληρωθεί, τα αποτελέσματα της σχεδίασης, αναλύονται για να παράξουν τη τελική διεπαφή. Τα παραγόμενα UIs συνήθως αποθηκεύονται σε UsiXML ενώ η εφαρμογή δίνει την επιλογή στο χρήστη να τα αποθηκεύσει και ως UIML.



Εικόνα 17: Παράδειγμα SketchiXML

Στις παρακάτω εικόνες φαίνεται η περιγραφή της διεπαφής σε UsiXML (Εικόνα 18) και σε UIML (Εικόνα 19).

```
<cuiModel id="Project_Name-cui" name="Project_Name-cui">
    <window id="window_0" name="window_0" isVisible="true"
        isEnabled="true" width="800" height="603"
        isAlwaysOnTop="false" isResizable="true">
        <gridBagBox id="Box_0" name="Box_0">
            <constraint gridx="5" gridy="4" gridwidth="5"
                gridheight="0" weightx="1" weighty="1" fill="none">
                <inputText id="TextField_0" name="TextField_0"
                    isVisible="true" isEnabled="true"
                    fgColor="#000000" bgColor="#ffffff"
                    textColor="#000000" maxLength="100"
                    numberOfColumns="20" numberOfLines="1"
                    isPassword="false" isWordWrapped="true"
                    forceWordWrapped="true" isEditable="true" defaultFilter=""/>
            </constraint>
            <constraint gridx="16" gridy="3" gridwidth="4"
                gridheight="2" weightx="1" weighty="1" fill="none"> <comboBox id="ComboBox_0" name="ComboBox_0" isVisible="true" isEnabled="true"
            </constraint>
            <constraint gridx="10" gridy="6" gridwidth="6" gridheight="2" weightx="1" weighty="1" fill="none">
                <inputText id="TextArea_0" name="TextArea_0"
                    isVisible="true" isEnabled="true"
                    fgColor="#000000" bgColor="#ffffff"
                    textColor="#000000" maxLength="100"
                    numberOfColumns="20" numberOfLines="0"
                    isPassword="false" isWordWrapped="true"
                    forceWordWrapped="true" isEditable="true" defaultFilter=""/>
            </constraint>
            <constraint gridx="9" gridy="9" gridwidth="5"
                gridheight="2" weightx="1" weighty="1" fill="none">
                <button id="Button_0" name="Button_0"
                    isVisible="true" isEnabled="true"
                    fgColor="#000000" bgColor="#ece9d8" textColor="#000000"/>
            </constraint>
        </gridBagBox>
    </window>
</cuiModel>
```

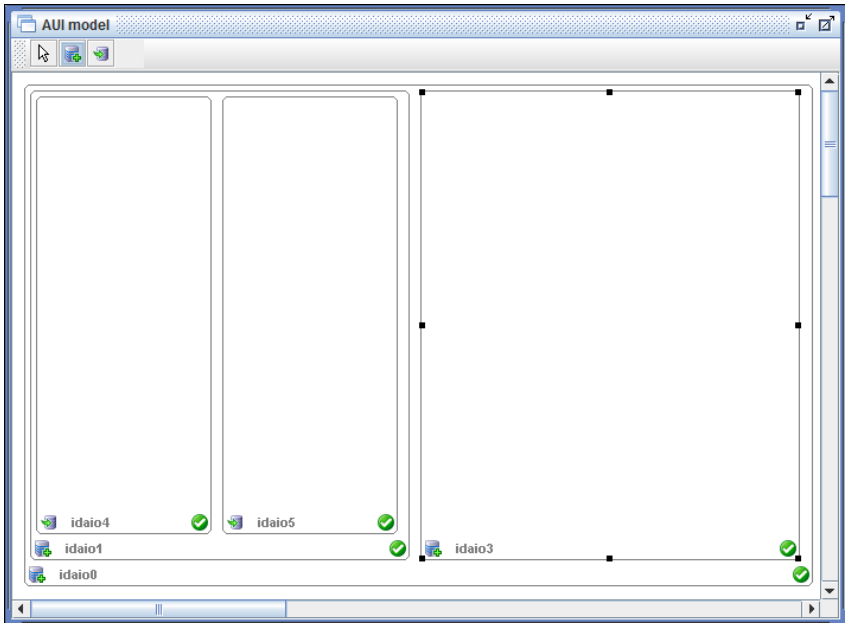
Εικόνα 18: Ο UsiXML κώδικας

```
<?xml version="1.0" encoding="UTF-8"?>
<uiml>
    <interface id="interface" how="replace" export="optional">
        <structure how="replace" export="optional">
            <part id="GridBag_Box_0" class="JFrame" where="last"
                how="replace" export="optional">
                <style how="replace" export="optional">
                    <property name="size" how="replace"
                        export="optional">800, 603</property>
                    <property name="layout" how="replace" export="optional">null</property>
                    <property name="resizable" how="replace" export="optional">true</property>
                    <property name="title" how="replace"
                        export="optional">Project Name</property>
                </style>
                <part id="TextField_0" class="JTextField" where="last" how="replace" export="optional">
                    <style how="replace" export="optional">
                        <property name="bounds" how="replace" export="optional">103,76,107,20</property>
                    </style>
                </part>
                <part id="TextField_0" class="JTextField" where="last" how="replace" export="optional">
                    <style how="replace" export="optional">
                        <property name="bounds" how="replace" export="optional">103,76,107,20</property>
                    </style>
                </part>
                <part id="ComboBox_0" class="JComboBox" where="last" how="replace" export="optional">
                    <style how="replace" export="optional">
                        <property name="text" how="replace"
                            export="optional">ComboBox 0</property>
                        <property name="bounds" how="replace" export="optional">316,63,89,43</property>
                    </style>
                </part>
                <part id="TextArea_0" class="JTextArea" where="last" how="replace" export="optional">
                    <style how="replace" export="optional">
                        <property name="text" how="replace"
                            export="optional">TextArea 0</property>
                        <property name="bounds" how="replace" export="optional">202,113,122,50</property>
                    </style>
                </part>
                <part id="Button_0" class="JButton" where="last" how="replace" export="optional">
                    <style how="replace" export="optional">
                        <property name="text" how="replace"
                            export="optional">Button 0</property>
                        <property name="bounds" how="replace" export="optional">174,186,98,42</property>
                    </style>
                </part>
            </part>
        </structure>
```

Εικόνα 19: Ο UIML κώδικας

IdealXML

Το ideal XML είναι ένα εργαλείο για τη σχεδίαση διεπαφών ανεξαρτήτως εκτός περιβάλλοντος χρήσης και modality (modality-independent). Για να το επιτύχει αυτό βασίζεται το AUI επίπεδο αφαίρεσης όπως αυτό προσδιορίζεται από το Cameleon Reference Framework. Οι σχεδιαζόμενες διεπαφές αποτελούνται από αφηρημένους containers (Abstract Containers: AC) όπως και από αφηρημένα διαδραστικά αντικείμενα (Abstract Interaction Objects: AIOs). Το τελικώς παραγόμενο μοντέλο μπορεί είτε να εκτελεστεί μέσω κατάλληλων μετασχηματισμών σε κώδικα είτε να μετασχηματιστεί στο πιο concrete επίπεδο σχεδίασης (CUI), ώστε να προσδιοριστούν περεταίρω λεπτομέρειες.



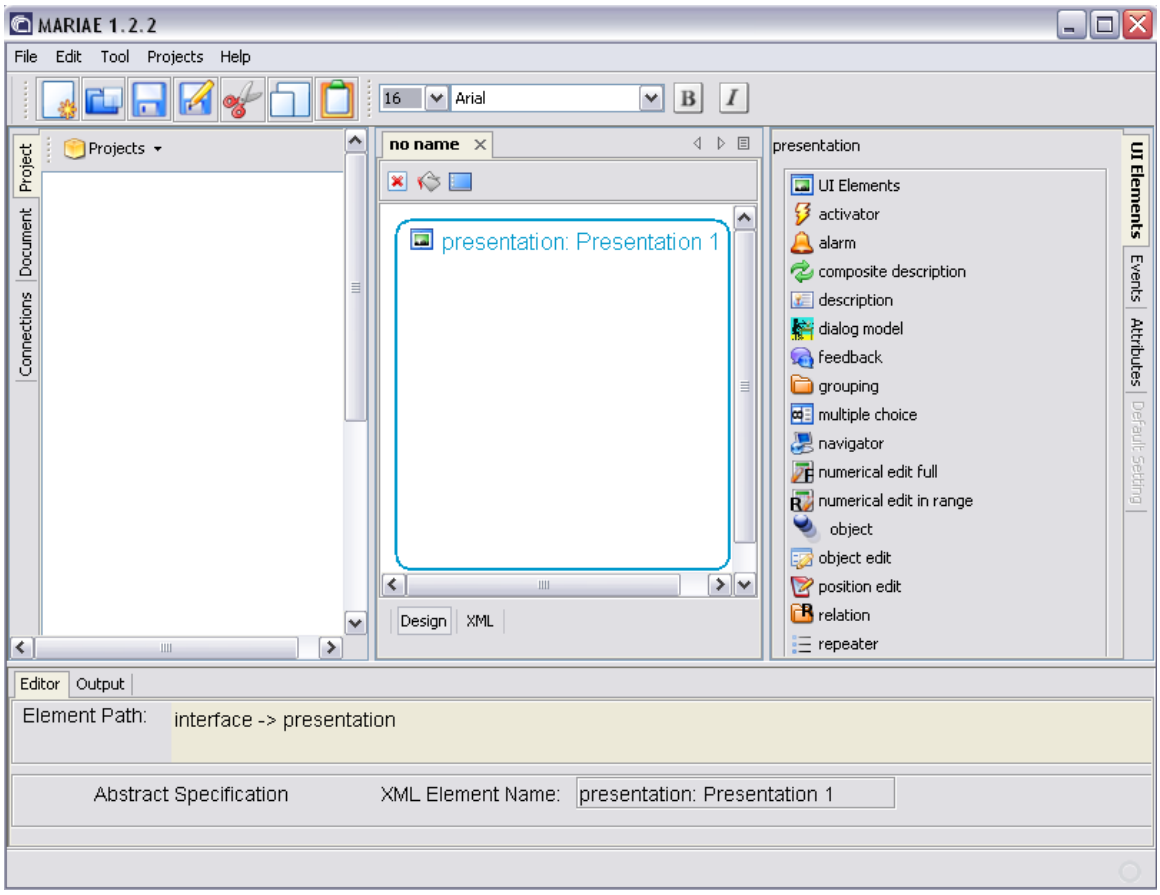

```
<?xml version="1.0" encoding="UTF-8"?>
<auimodel>
    <abstractContainer id="idaio0" name="idaio0">
        <abstractContainer id="idaio1" name="idaio1">
            <abstractIndividualComponent id="idaio4" name="idaio4">
            </abstractIndividualComponent>
            <abstractIndividualComponent id="idaio5" name="idaio5">
            </abstractIndividualComponent>
        </abstractContainer>
        <abstractContainer id="idaio3" name="idaio3">
        </abstractContainer>
    </abstractContainer>
</auimodel>
```

Εικόνα 20: Παραγόμενο AUI Μοντέλο

Multimodal Teresa- MARIAE

Το Multimodal TERESA, είναι ένα εργαλείο το οποίο βοηθάει στη δημιουργία διεπαφών για πλατφόρμες διαφορετικού τύπου (desktop, mobile, vocal, multimodal, digital TV), αρχίζοντας από τη λογική περιγραφή των

διεπαφών. Είναι σε θέση να παράγει διεπαφές που προσαρμόζονται στους πόρους που είναι διαθέσιμοι ανάλογα με ποιά γλώσσα θα διαλέξουμε να κάνουμε την υλοποίηση. Το tool αυτό έχει πλέον αντικατασταθεί από το MARIAE. Το MARIAE παρέχει μια νέα λύση η οποία του επιτρέπει να εκμεταλλευτεί τα task models (που αναπαριστώνται στα ConcurTaskTrees) και τα users interfaces (τα οποία έχουν γραφτεί σε MARIAXML) για το σχεδιασμό και την ανάπτυξη διακρατικών διεπαφών βασιζόμενες σε Web υπηρεσίες για διαφορετικές πλατφόρμες. Το εργαλείο είναι σε θέση να εισάγει αυτόματα υπηρεσίες και περιγραφές και να υποστηρίξει διακρατικές διασυνδέσεις βασικών εργασιών με διαδικτυακές δραστηριότητες. Επιπλέον με μια σειρά ημι-αυτόματες μετατροπές είναι σε θέση να αξιοποιήσει τις πληροφορίες σε αυτές τις υπηρεσίες και διασυνδέσεις για να παράξει διεπαφές για διαφορετικές πλατφόρμες.

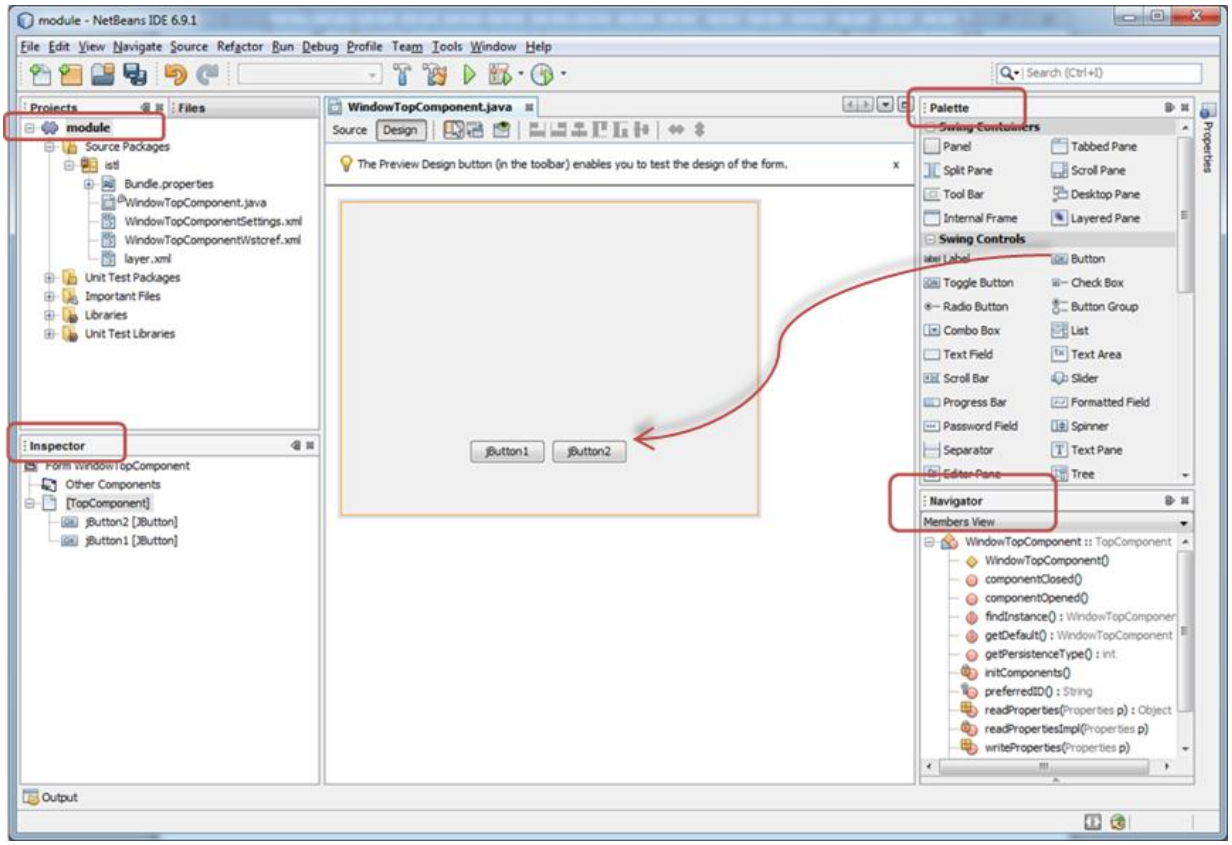


Εικόνα 21: Περιβάλλον χρήσης MARIAE.

4. Προσέγγιση

Δεδομένου του πλήθους των προσεγγίσεων για την ανάπτυξη-περιγραφή διεπαφών για πολλαπλά περιβάλλοντα που αναλύθηκαν στο προηγούμενο κεφάλαιο θεωρούμε ως καταλληλότερη την βασιζόμενη σε μοντέλα. Αυτό διότι δίνει τη δυνατότητα προσδιορισμού ενός κοινού-ενοιαίου κύκλου ανάπτυξης αλλά παράλληλα και τη δυνατότητα απομόνωσης της πολυπλοκότητας ανάπτυξης αυτών και εστίασης σε συγκεκριμένες λεπτομερείς που αφορούν το εκάστοτε επίπεδο αφαίρεσης. Επιπλέον οι γλώσσες περιγραφής διεπαφών σε ορισμένες περιπτώσεις ενσωματώνουν στη λογική τους τη δυνατότητα προσδιορισμού εναλλακτικών τελικών διεπαφών ανάλογα με το περιβάλλον χρήσης (context of use). Αυτό δίνει τη δυνατότητα οι σχεδιαζόμενες και παραγόμενες τελικώς διεπαφές να είναι προσαρμόσιμες τόσο με βάσει διαφοροποιήσεις στις τελικές πλατφόρμες, όσο στο περιβάλλον αλλά και στο στερεότυπο των εκάστοτε τελικών χρηστών. Η UsiXML που αποτελεί την υλοποίηση αναφοράς του Cameleon Reference Framework αποτελεί ιδανική επιλογή αφού συναθροίζει πλήθος στοιχείων που είναι απαραίτητα για την υποστήριξη διεπαφών για πολλαπλά περιβάλλοντα. Συγκεκριμένα, δίνει τη δυνατότητα προσδιορισμού πολυκεντρικών διεπαφών (multimodal), μέσω του AUI στο οποίο περιγράφεται με τρόπο ανεξάρτητο καναλιού αλληλεπίδρασης μια διεπαφή. Επίσης εξίσου καθοριστικός παράγοντας για την επιλογής της είναι το γεγονός ότι υιοθετεί ένα χρηστικό-κεντρικό στυλ ανάπτυξης το οποίο διευκολύνει την ανάλυση και εν γένει την ανάπτυξη των παραγόμενων διεπαφών. Επιπλέον παρέχει την ευελιξία εκκίνησης της σχεδίασης διεπαφής από οποιοδήποτε επίπεδο αφαίρεσης(multi-path development). Το επίπεδο αφαίρεσης με το οποίο ασχολείται η συγκεκριμένη πτυχιακή είναι αυτό του CUI(concrete user interface) έτσι ώστε να δοθεί η δυνατότητα σχεδίασης διεπαφών απο μια πληθώρα(προκαθορισμένων η και νέων) αντικειμένων βασισμένων σε ένα συγκεκριμένο κανάλι αλληλεπίδρασης.

Για την υποστήριξη της συγκεκριμένης φάσης σχεδίασης μιας διεπαφής επιλέχθηκε το NetBeans Platform API δίνοντας έτσι την δυνατότητα επεκτασιμότητας της όλης εφαρμογής ώστε να καλύψει και τα υπόλοιπα επίπεδα σχεδίασης (CTT,AUI). Η επιλογή του έγινε στα πλαίσια της εξειδικευμένης παρεχόμενης και σχετικά καλά τεκμηριωμένης λειτουργικότητάς του υπό μορφή υπό-μονάδων (modules) παρέχοντας υποστήριξη πλήθους καθηκόντων όπως drag & drop, παροχή γραφικών και μη αντικειμένων για την υποστήριξη πλοήγησης σε ιεραρχικά δεδομένα, και πολλά άλλα. Επίσης θεμελιώδες πλεονέκτημα αυτού είναι το γεγονός πως ο πηγαίος κώδικάς του είναι διαθέσιμος κάτω από άδειες λογισμικό ανοιχτού κώδικα. Ένα άλλο σημαντικό πλεονέκτημα που προσφέρει το NetBeans platform είναι το γεγονός πως στηρίζεται στη λογική των modules. Κάθε module αποτελείται από ένα σύνολο Java κλάσεων συνοδευόμενο από κατάλληλα xml αρχεία απαραίτητα για την ορθή ενσωμάτωση και εύρυθμη λειτουργία των (π.χ. dock/undock windows) στα πλαίσια της πλατφόρμας. Κάθε συμβατό module μπορεί να εγκατασταθεί στο εν λόγω IDE και να θεωρηθεί ως επέκταση της πλατφόρμας ενώ ακόμα δίνει τη δυνατότητα σε κάποιον τρίτο χρήστη αφού εγκαταστήσει την επέκταση αυτή να την επεκτείνει προσθέτοντας της περαιτέρω λειτουργίες χωρίς ωστόσο να χρειάζεται να επέμβει στον κώδικα του module το οποίο επιθυμεί να επεκτείνει.



Εικόνα 22: Λειτουργικότητα NetBeans

Ο παρακάτω πίνακας συνοψίζει τις επιλογές που έγιναν σε επίπεδο προσέγγισης συναρτήσει της ουσιαστικής συμβολής των.

Σχεδιαστικές επιλογές	Κληροδοτηθέντα Πλεονεκτήματα
NetBeans	Η πλατφόρμα του NetBeans η οποία είναι η βάση της εφαρμογής.
UsiXML	Η UsiXML είναι η γλώσσα περιγραφής των διεπαφών που δημιουργούνται από το εργαλείο.
Matisse	Με την χρήση του Matisse έγινε ο σχεδιασμός της όψης του σχεδιασμού της διεπαφής.

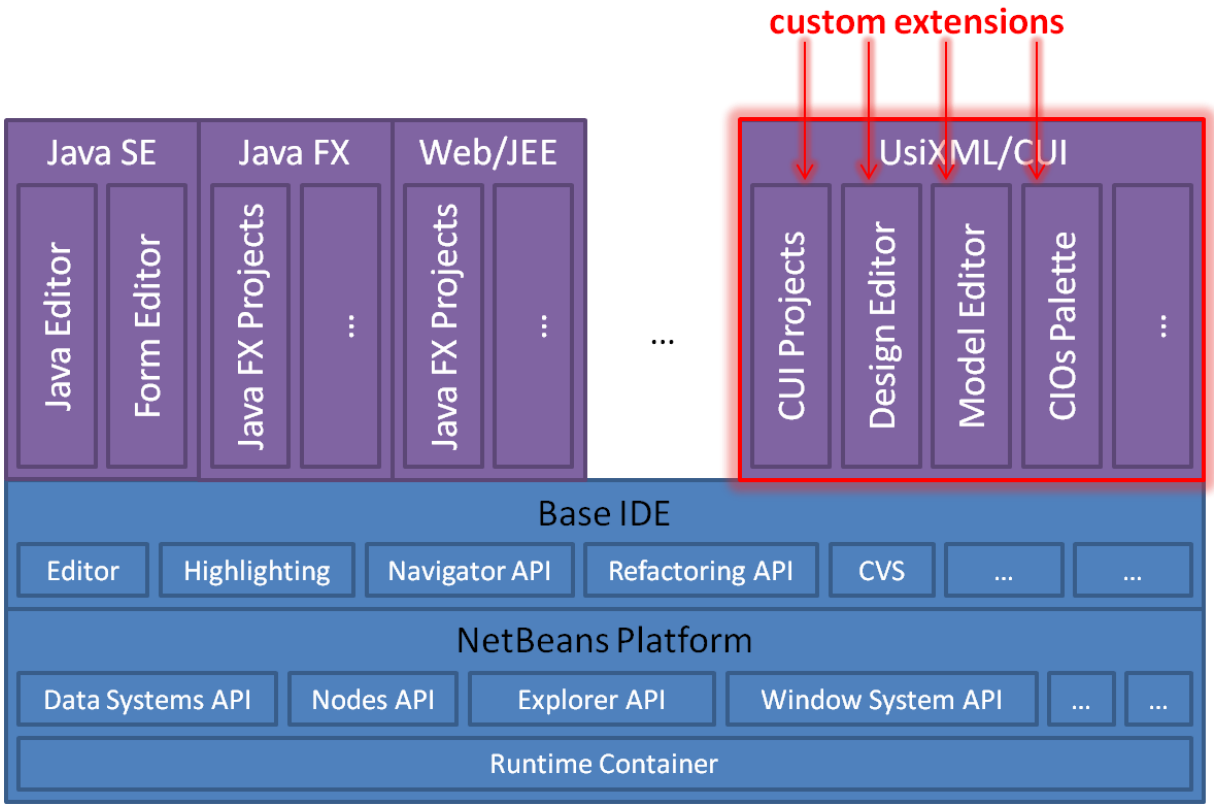
Πίνακας 1: Προσσέγγιση Σχεδιασμού Εφαρμογής

5. Υλοποίηση

Στα πλαίσια αυτής της πτυχιακής αναπτύχθηκε ένα εργαλείο ως επέκταση του NetBeans IDE το οποίο επιτρέπει στους χρήστες να δημιουργούν διεπαφές για πολλαπλά περιβάλλοντα. Η επέκταση βασίστηκε σε θεμελιώδη APIs του NetBeans τα οποία χρησιμοποιήθηκαν για την λειτουργία και τον συντονισμό των στοιχείων της εφαρμογής .

5.1 Αρχιτεκτονική

Στην εικόνα που ακολουθεί παρουσιάζεται η αρχιτεκτονική της πλατφόρμας του NetBeans και τα APIs στα οποία βασίζεται η εφαρμογή, παρουσιάζονται επίσης τα επιμέρους εργαλεία (Αναγνώριση τύπων εφαρμογής, σχεδίαση διεπαφής, βιβλιοθήκη αντικειμένων κ.α.) τα οποία αναπτύχθηκαν για να υποστηρίξουν τη λειτουργία της εφαρμογής. Τα μπλε κουτιά αναφέρονται στα βασικά APIs του NetBeans platform πάνω στα οποία στηρίζονται άλλα ήδη υλοποιημένα και ενσωματωμένα στο NetBeans IDE plug-ins όπως το Java SE, Java FX κα. Πάνω σε αυτά τα APIs δημιουργούνται και τα custom plug-ins (UsiXML/CUI) για να είναι συμβατά με την πλατφόρμα. Τα μωβ κουτιά αναπαριστούν είναι ήδη υλοποιημένα plug-ins ενώ μέσα περιέχουν τις λειτουργίες που προσφέρουν. Για παράδειγμα το UsiXML/CUI του οποίου τα επιμέρους εργαλεία θα αναλυθούν στη συνέχεια αποτελείται από τον τύπο project, το Design Editor, την παλέτα των αντικειμένων κι άλλα επιμέρους εργαλεία.

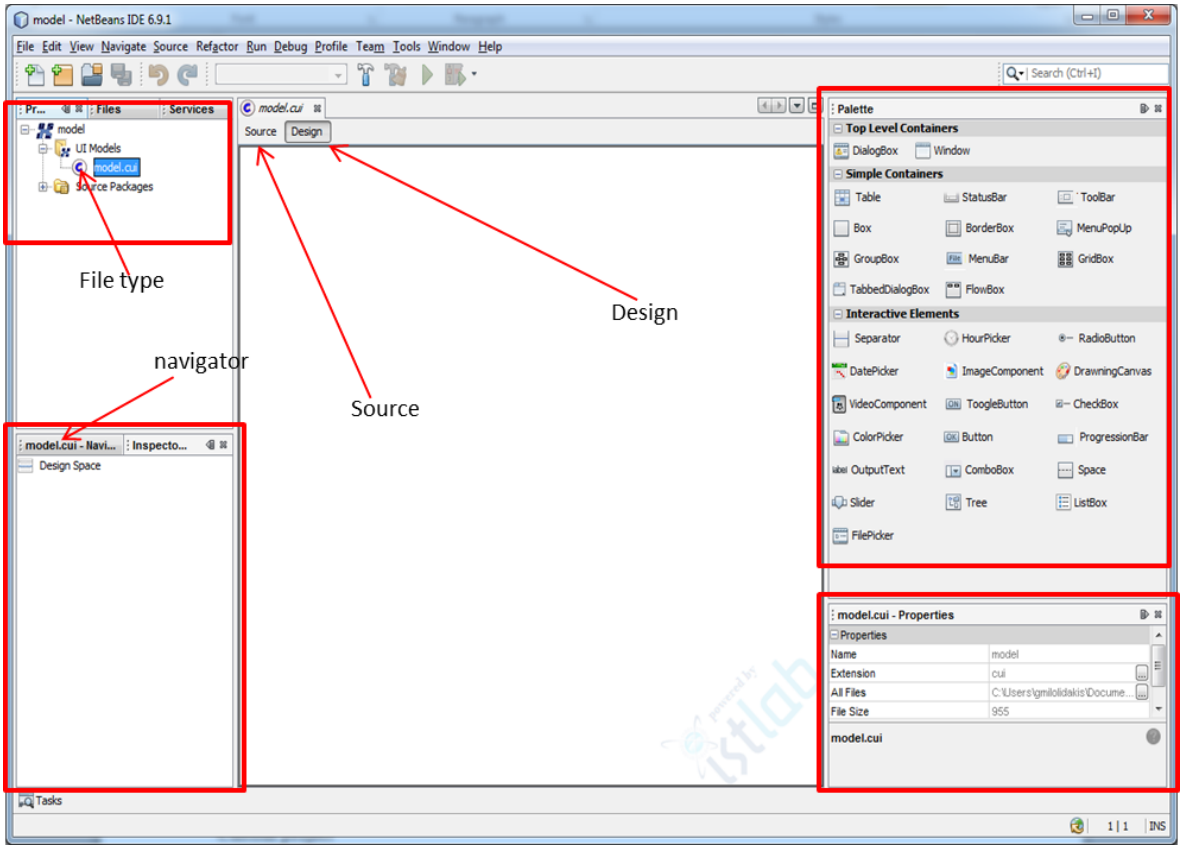


Εικόνα 23: Αρχιτεκτονική Πλατφόρμας και Εφαρμογής

5.2 Εργαλείο (Tool)

Υλοποιώντας το εργαλείο βασιζόμενοι στα δημόσια APIs του NetBeans το εργαλείο αποκτά όλα τα χαρακτηριστικά και τις δυνατότητες που του προσφέρουν αυτά τα APIs, ακόμα μετά την εγκατάσταση του θεωρείται ως μέρος της κύριας πλατφόρμας η οποία πλέον αναγνωρίζει τα UsiXML projects καθώς και τα cui αρχεία τα οποία περιέχονται σε αυτά (Εικόνα 24). Έτσι ο χρήστης μπορεί να δημιουργεί UsiXML εφαρμογές οι οποίες περιέχουν cui αρχεία για την δημιουργία διεπαφών. Τα project αυτά έχουν όλες τις βασικές ιδιότητες που τους παρέχει η πλατφόρμα όπως Save, Delete κτλ.

Στην Εικόνα 24 εκτός απο το τύπο αρχείου και τον τύπο project φαίνονται επίσης και ο οπτικός επεξεργαστής ο οποίος είναι χωρισμένος σε δυο όψεις μια για την σχεδίαση της διεπαφής και μία για την παρουσίαση της UsiXML μορφής στην οποία αποθηκεύεται, η παλέτα με τα διαθέσιμα αντικείμενα τα οποία ο χρήστης τραβά στη σκηνή, ο επεξεργαστής ιδιοτήτων των αντικειμένων, η ιεραρχική απεικόνιση των αντικειμένων και ο πλοηγός.

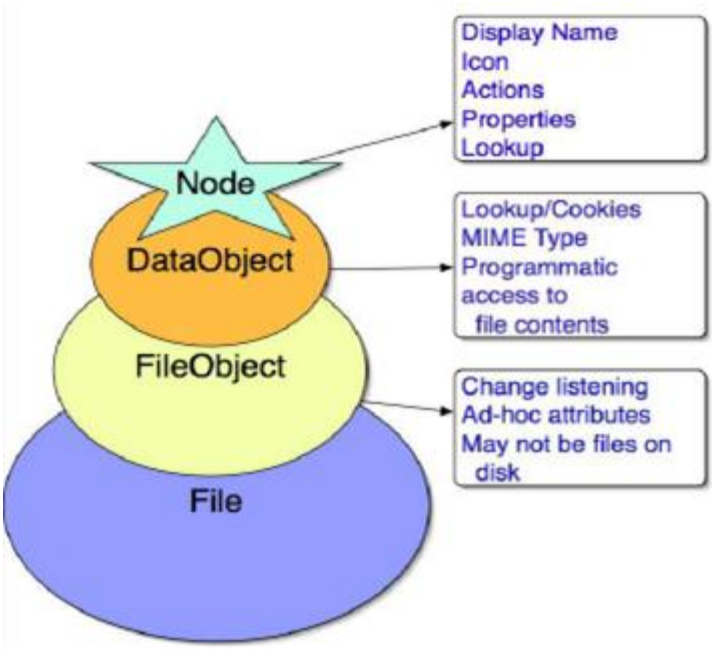


Εικόνα 24: Η εφαρμογή ως μέρος της πλατφόρμας του NetBeans

5.2.1 Ορισμός Τύπου Αρχείων (File Type extension)

Για τις ανάγκες της εφαρμογής δημιουργήθηκε ένας τύπος αρχείου με επέκταση ".cui" ο οποίος αποτελείται από τέσσερα επίπεδα Στο πρώτο επίπεδο (file) αποθηκεύεται η πληροφορία, ο UsiXML κώδικας. Το δεύτερο επίπεδο (FileObject) παρέχει πληροφορίες για το αρχείο όπως το path στο οποίο βρίσκεται, το όνομα του κτλ. Το FileObject είναι απλά ένας container για την πληροφορία, ο οποίος δεν χρειάζεται να γνωρίζει το τύπο της πληροφορίας που περιέχει.

Αντιθέτως το dataObject, το οποίο αναφέρεται στο fileObject γνωρίζει τον τύπο του αρχείου το οποίο αναπαριστά. Έτσι μέσω του dataObject γίνεται η επικοινωνία του χώρου σχεδίασης με το αρχείο, αφού είναι αυτό που υποστηρίζει τη δημιουργία των δυο όψεων ενώ επίσης μέσω αυτού γίνεται η αποθήκευση της εφαρμογής. Το τελευταίο επίπεδο (node) παρέχει τη γραφική απεικόνιση του αρχείου στο περιβάλλον του NetBeans, δηλαδή καθορίζει ποιο θα είναι το εικονίδιο το οποίο θα αναπαριστά το cui αρχείο ως γνωστό τύπο αρχείων. Στην πράξη το cui αρχείο είναι μια επέκταση της XML την οποία υποστηρίζει το NetBeans δηλαδή (xml+.cui). Συνεπώς σε περίπτωση που επιχειρήσουμε να ανοικουμε αυτό το αρχείο μέσω μιας άλλης εφαρμογής η οποία δεν γνωρίζει αυτήν την επέκταση, το αρχείο θα συμπεριφερθεί σαν τυπικό xml αρχείο.

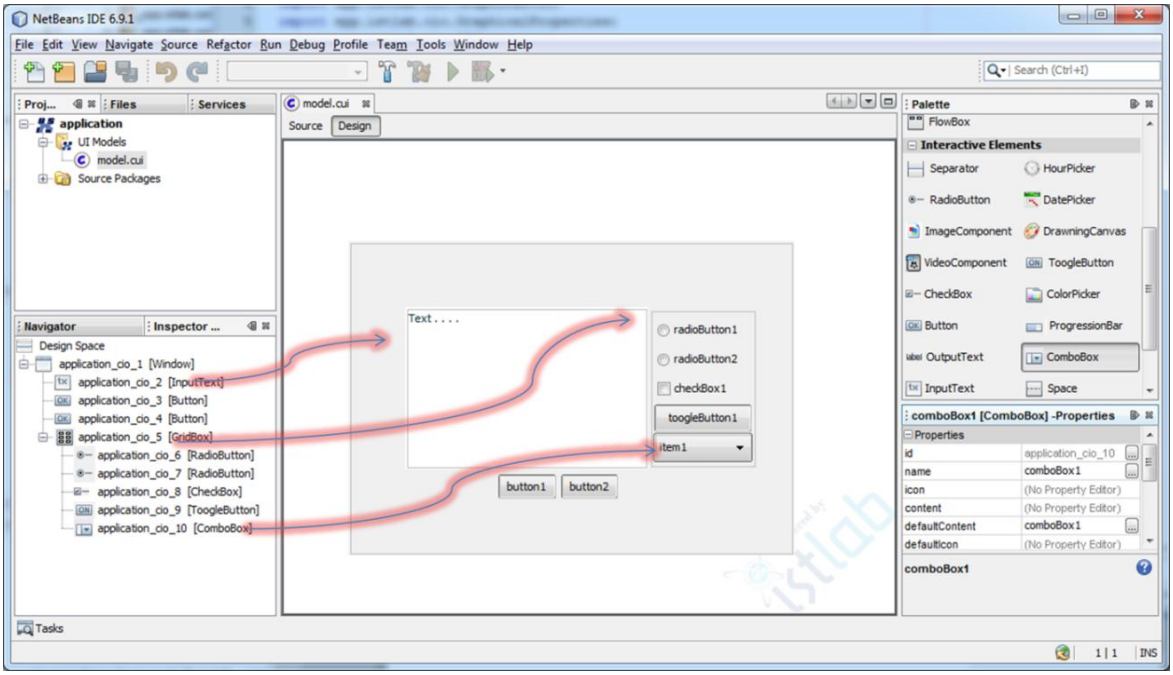


Εικόνα 25: Ορισμός τύπου αρχείων

5.2.2 Οπτικός Επεξεργαστής (Visual Editor)

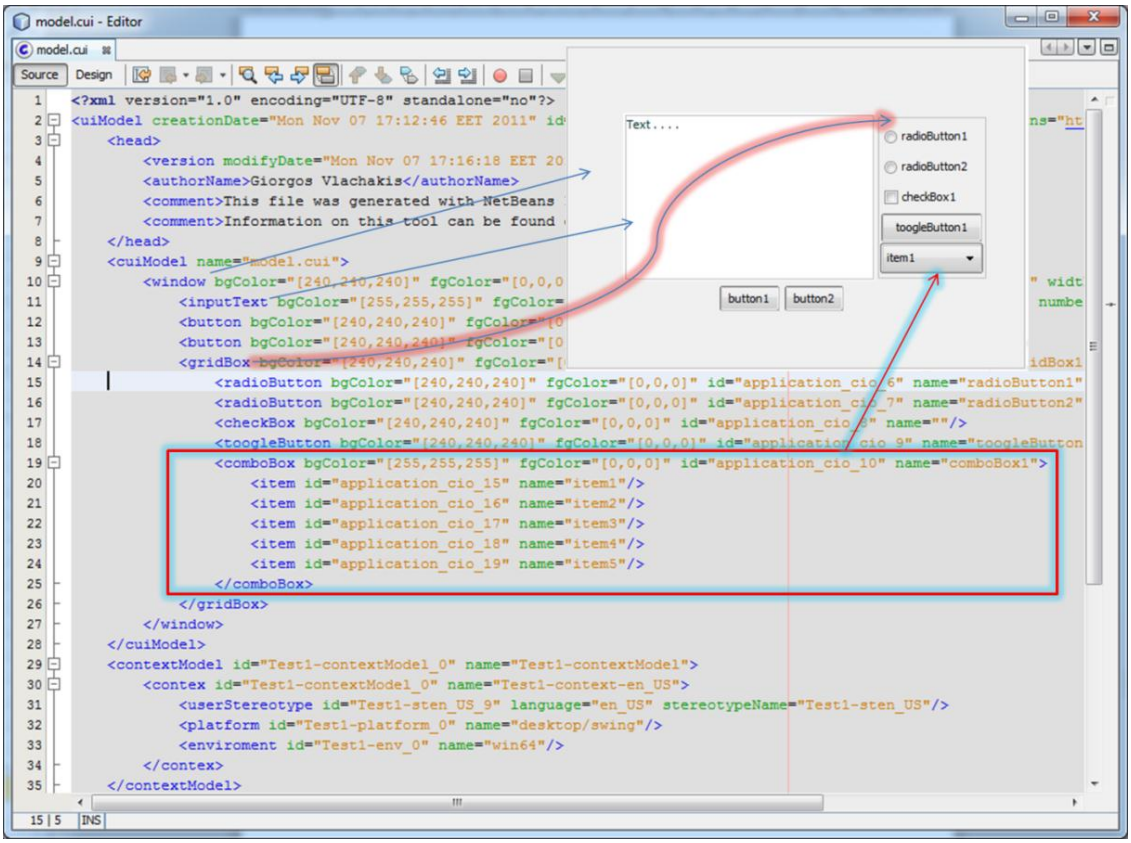
Ο Editor είναι ο χώρος στον οποίο δημιουργούμε τη διεπαφή. Ο editor είναι άμεσα συνδεδεμένος με το file type, (file type χρησιμοποιεί τον Visual Editor για την δημιουργία των όψεων) ουσιαστικά είναι η μετάφραση του cui αρχείου από UsiXML σε γραφική διεπαφή.

Στην παρούσα εφαρμογή παρέχονται δυο όψεις, DesignView (Εικόνα 26) η οποία είναι ο γραφικός editor όπου γίνεται η σύνθεση της εφαρμογής και SourceView όπου ο χρήστης βλέπει το UsiXML κώδικα που χρησιμοποιείται για την περιγραφή της διεπαφής. Για την δημιουργία της διπλής όψης του αρχείου χρησιμοποιήθηκε το module XMLMulitview για να συνδέσει το αρχείο με το γραφικό περιβάλλον, το οποίο όμως δεν είναι μέρος του επίσημου Netbeans API.



Εικόνα 26: Περιβάλλον σχεδίασης (Design View)

Στο Source View (Εικόνα 27) ολόκληρη η εφαρμογή μεταφράζεται σε UsiXML κώδικα, στον οποίο κάθε αντικείμενο της διεπαφής αποτυπώνεται σαν ένα XML Element με κάποιες βασικές ιδιότητες, ενώ η δομή του UsiXML αρχείου εξαρτάται από την ιεραρχία των αντικειμένων της διεπαφής μέσα στο Design Space. Εκτός από τα αντικείμενα της διεπαφής ο πηγαίος κώδικας κρατά επιπλέον πληροφορίες όπως τον δημιουργό της εφαρμογής, την ημερομηνία δημιουργίας και το λειτουργικό στο οποίο δημιουργήθηκε.



Εικόνα 27: Πηγαίος κώδικας (Source View)

5.2.3 Παλέτα (Palette)

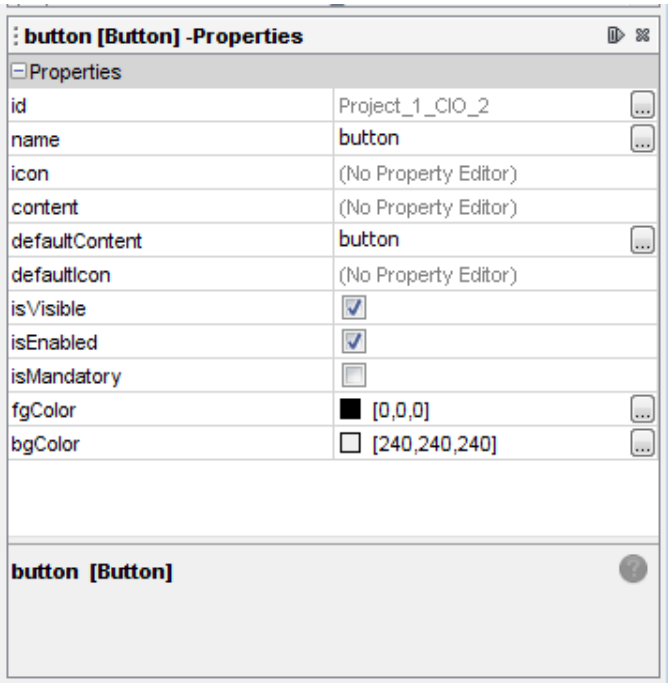
Η παλέτα βρίσκεται στα δεξιά του editor περιέχει τα διαθέσιμα αντικείμενα για την σχεδίαση των διεπαφών. Κάθε αντικείμενο που χρησιμοποιείται για την δημιουργία μιας διεπαφής είναι ένα σύνολο τριών διαφορετικών αντικειμένων που το κάθε ένα από αυτά έχει διαφορετικά καθήκοντα. Πρώτο είναι το αντικείμενο παλέτας. Περιγράφεται σε μια κλάση η οποία χρησιμοποιείται για να ομαδοποιήσει όλα τα γραφικά αντικείμενα τα οποία φαίνονται στη παλέτα και να τους δώσει κάποιες λειτουργίες, όπως το drag & drop για την επιθυμητή συμπεριφορά τους. Δεύτερο είναι το γραφικό αντικείμενο, δηλαδή αυτό που βλέπει ο χρήστης κατά τη σύνθεση μιας διεπαφής (button, slider κα.), το οποίο βασίζεται στις ήδη υλοποιημένες Java κλάσεις οι οποίες παρέχουν αυτά τα αντικείμενα (JButton, JSlider κα.) . Το τρίτο αντικείμενο είναι το αντικείμενο μοντέλο, ο χρήστης αγνοεί την ύπαρξη του καθώς δεν είναι ορατό στη διαδικασία σχεδίασης αλλά υπάρχει εκεί για να κρατά τις ιδιότητες του γραφικού αντικειμένου με σκοπό να δημιουργεί τον UsiXML κώδικα και να επικοινωνεί με τον Επεξεργαστή ιδιοτήτων για την προβολή των παραμέτρων του κάθε αντικειμένου. Τα δυο αυτά αντικείμενα είναι συγχρονισμένα μεταξύ τους έτσι ώστε αλλαγές που γίνονται στο ένα να γίνονται αυτόματα και στο άλλο για να μην υπάρχει κίνδυνος να χαθούν δεδομένα. Η σύνδεση των τριών αυτών αντικειμένων γίνεται σε ένα xml αρχείο το οποίο περιέχει πληροφορίες για την αρχιτεκτονική όλης της εφαρμογής, των παραθύρων που χρησιμοποιούνται, όπως και τις κατηγορίες που περιέχει η παλέτα και ποια είναι η γραφική απεικόνιση του κάθε αντικειμένου. Έτσι με την χρήση αυτών των τριών αντικειμένων μπορούμε να φτιάξουμε οποιοδήποτε (custom) αντικείμενο επιθυμούμε και να το τοποθετήσουμε στη παλέτα για να το χρησιμοποιήσουμε.

Στην παρακάτω εικόνα φαίνεται η περιγραφή των αντικειμένων, και συγκεκριμένα του Button στο XML αρχείο καθώς και το πως τοποθετούνται τα αντικείμενα στη παλέτα. Το Button βρίσκεται στο φάκελο "Interactive Elements" ο οποίος είναι ένας απο τους τρείς φακέλους που αντιστοιχούν στις κατηγορίες της παλέτας. Ως γραφικό αντικείμενο "component" είναι ένα στιγμιότυπο της κλάσης GButton και ως αντικείμενο μοντέλο "Bean" ένα στιγμιότυπο της κλάσης Button. Άλλες πληροφορίες που υπάρχουν είναι η διαδρομή (path) στην οποία βρίσκεται το εικονίδιο που μπαίνει στη παλέτα καθώς και το κείμενο που θα το συνοδεύει.

Εικόνα 28: Περιγραφή αντικειμένου παλέτας

5.2.4 Επεξεργαστής Ιδιοτήτων (Properties Editor)

Ο property editor (Εικόνα 29) προβάλει τις ιδιότητες όποιου αντικειμένου επιλέξουμε. Για να λειτουργήσει ο property editor υλοποιήθηκε μία κλάση η οποία περιέχει ένα πίνακα με τις ιδιότητες για κάθε τύπο αντικειμένου οποία συγχρονίζεται με το αντικείμενο μοντέλο το οποίο κρατά τις τιμές των ιδιοτήτων και η οποία με τη σειρά της ενημερώνει το γραφικό αντικείμενο. Όμως επειδή το γραφικό αντικείμενο είναι εξίσου συγχρονισμένο με το αντικείμενο μοντέλο αν αλλάξει μία ιδιότητα απευθείας στο γραφικό αντικείμενο (για παράδειγμα αν του αλλάξουμε το μέγεθος ενός αντικειμένου σέρνοντας το με το ποντίκι) τότε η αλλαγή αυτή θα γίνει και στον πίνακα ιδιοτήτων του αντικειμένου. Οι πληροφορίες που κρατούνται στον property editor είναι το id, το όνομα το μέγεθος του αντικειμένου κ.α. Επίσης στο πάνω μέρος του υπάρχει ένας τίτλος ο οποίος είναι το όνομα του αντικειμένου, ο τύπος του και η ένδειξη "Properties".

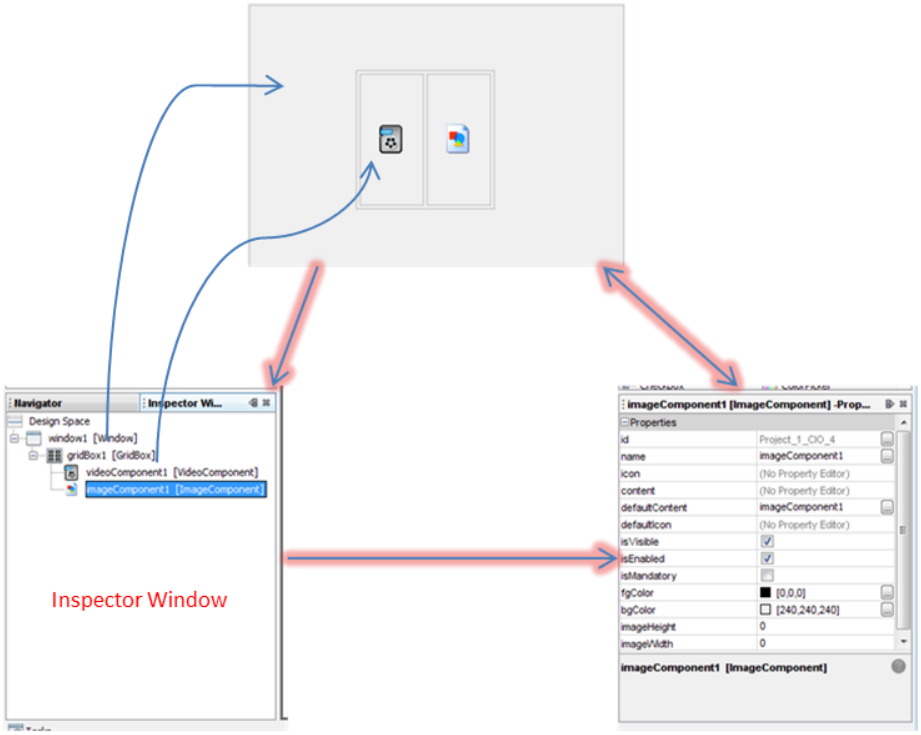


Εικόνα 29: Property Editor

5.2.5 Ιεραρχία Αντικειμένων (Inspector)

Ο inspector κρατάει την ιεραρχική δομή της διεπαφής. Διαχωρίζει τα αντικείμενα σε components και containers και μας δείχνει τη σχέση μεταξύ των αντικειμένων που έχουμε τοποθετήσει. Για την λειτουργία του υλοποιήθηκε ένα αντικείμενο, χρησιμοποιώντας το Nodes API module του NetBeans κι ένας TopComponent ο οποίος χρησιμοποιεί το Explorer & Property Sheet API module. Έτσι κάθε φορά που τοποθετείται η αφαιρείται ένα αντικείμενο από τη διεπαφή ο inspector δημιουργεί η αφαιρεί το αντίστοιχο node από τη δομή του. Τέλος ο inspector είναι συγχρονισμένος με τον property editor έτσι ώστε κάθε φορά που επιλέγω ένα αντικείμενο να εμφανίζονται αυτόματα και τα properties του.

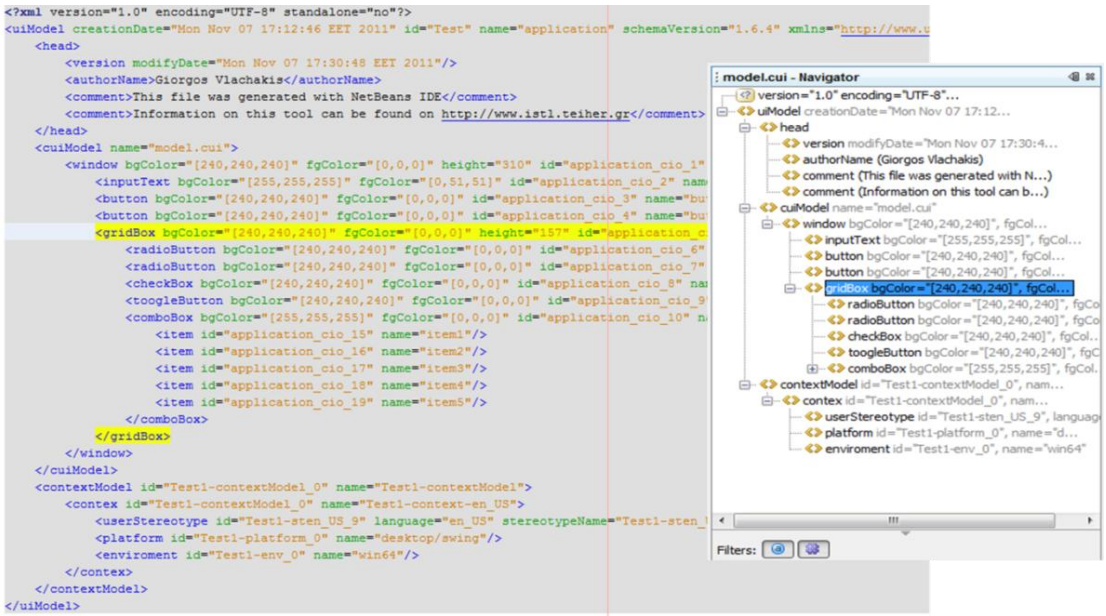
Στην εικόνα που ακολουθεί φαίνεται μια διεπαφή, οποία αποτελείται απο ένα παράθυρο, κι ένα κουτί με Grid Layout το οποίο περιέχει ένα μία εικόνα κι ένα βίντεο. Επίσης φαίνεται η αποτύπωση αυτής της διεπαφής στον inspector και η αλληλεπίδραση του inspector με τον Επεξεργαστή ιδιοτήτων.



Εικόνα 30: Inspector

5.2.6 Πλοήγηση (Navigator)

Ο navigator είναι συνδεδεμένος με το SourceView, βοηθάει στην πλοήγηση στον UsiXML κώδικα που δημιουργείται και παρουσιάζει τα elements και την δομή του XML. Έτσι όπως φαίνεται και στην Εικόνα 31 επιλέγοντας ένα αντικείμενο στον navigator χρωματίζεται το κομμάτι στο οποίο γίνεται η περιγραφή του στο Source View.



Εικόνα 31: Navigator

5.2.7 Ορισμός Τύπου Εφαρμογής (Project Type extension)

Για την υποστήριξη ενός τύπου εφαρμογής δημιουργήθηκε ένα σύνολο κλάσεων για να υποστηρίξουν τη λογική εμφάνιση και την ιεραρχική δομή των φακέλων έτσι ώστε να αναγνωρίζονται από το περιβάλλον του NetBeans ως ένας γνωστός τύπος εφαρμογής. Αφού δημιουργήθηκαν όλα τα μέρη της εφαρμογής υλοποιήθηκε ένα project Template όπως παρέχεται από το NetBeans το οποίο δίνει τη δυνατότητα στο χρήστη να φτιάξει ένα οδηγό δημιουργίας project (UsiXML project).

5.3 Συγκεντρωτική παρουσίαση επιμέρους λειτουργιών

Ο παρακάτω πίνακας συνοψίζει όλα τα επιμέρους εργαλεία τα οποία αναλύθηκαν στις προηγούμενες παραγράφους. Τα εργαλεία αυτά είναι βασικά για την υλοποίηση της UsiXML επέκτασης του NetBeans IDE αλλά υπάρχουν κι άλλα εργαλεία τα οποία βοηθούν στην αλλιλεπίδραση του χρήστη με την εφαρμογή κατά τη φάση της σχεδίασης, και προσφέρουν λειτουργίες όπως η μετακίνηση του αντικειμένου μέσα στη διεπαφή, η απομάκρυνσή του από αυτή η αλλαγή του μεγέθους του κτλ. Επίσης για όλα τα προσφερόμενα αντικείμενα στα πλαίσια αυτής της επέκτασης δημιουργήθηκαν νέες κλάσεις επεκτείνοντας τις ήδη υλοποιημένες Java κλάσεις που υπάρχουν για τη δημιουργία αυτών των αντικειμένων (πχ JButton, JPanel κτλ).

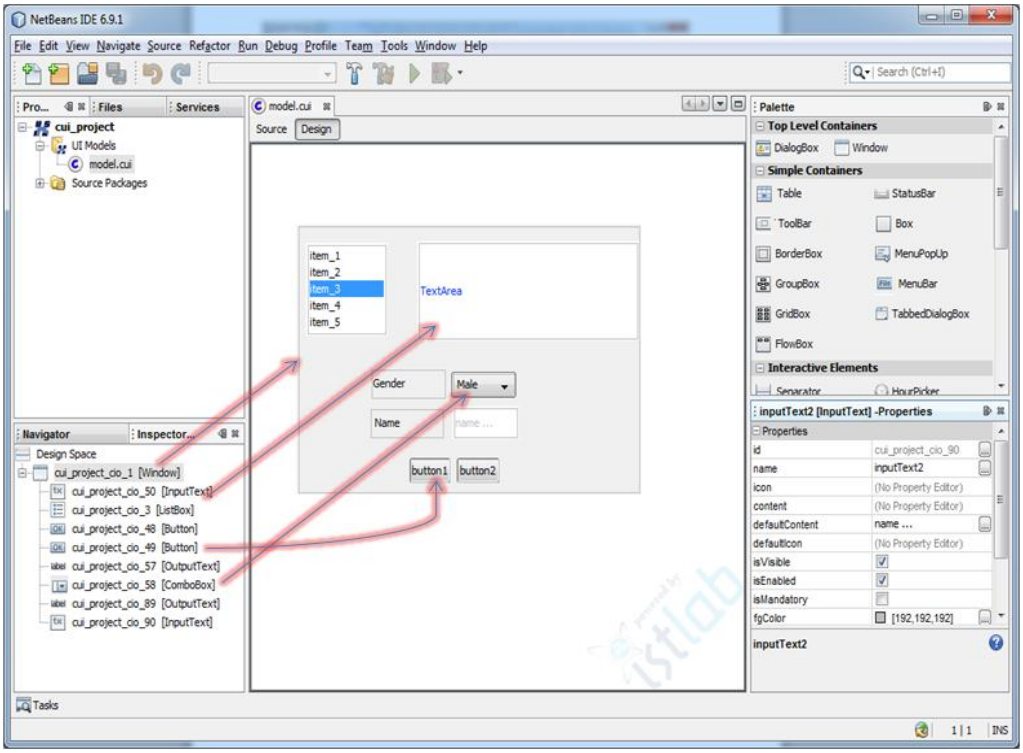
NetBeans Εργαλείο	Υπάρχουσες λειτουργίες NetBeans	Νέες Λειτουργίες που αναπτύθηκαν
File Type	Java files, XML files, C/C++, JSON.	CUI file. Δημιουργία νέου τύπου αρχείου.
Project Type	Java, PHP, C/C++ projects	USIXML project. Δημιουργία νέου τύπου project.
Visual Editor	Υποστηρίζει δυο όψεις (Matisse): a) Όψη σχεδίασης b) Όψη Java κώδικα.	Υποστηρίζει δυο όψεις: a) Όψη σχεδίασης (υπάρχουσα) b) Όψη UsiXML κώδικα. (νέα)
Palette	Περιέχει αντικείμενα για την σχεδίαση διεπαφών.	Εμπλουτισμός παλέτας με UsiXML αντικείμενα.
Properties Editor	Προβάλει και τροποποιεί τις ιδιότητες του κάθε γραφικού αντικειμένου.	Εμπλουτισμός έτσι ώστε να είναι συμβατός με τα UsiXML αντικείμενα.
Inspector	Παρουσιάζει την ιεραρχική δομή μιας διεπαφής.	Εμπλουτισμός ιεραρχικής δομής μιας διεπαφής με UsiXML artifacts.
Navigator	Λειτουργεί σα χάρτης για τον Java κώδικα.	Λειτουργεί σα χάρτης για τον UsiXML κώδικα.

Πίνακας 2: Συγκεντρωτική παρουσίαση επιμέρους εργαλείων.

5.4 Ενδεικτικό Παράδειγμα Χρήσης

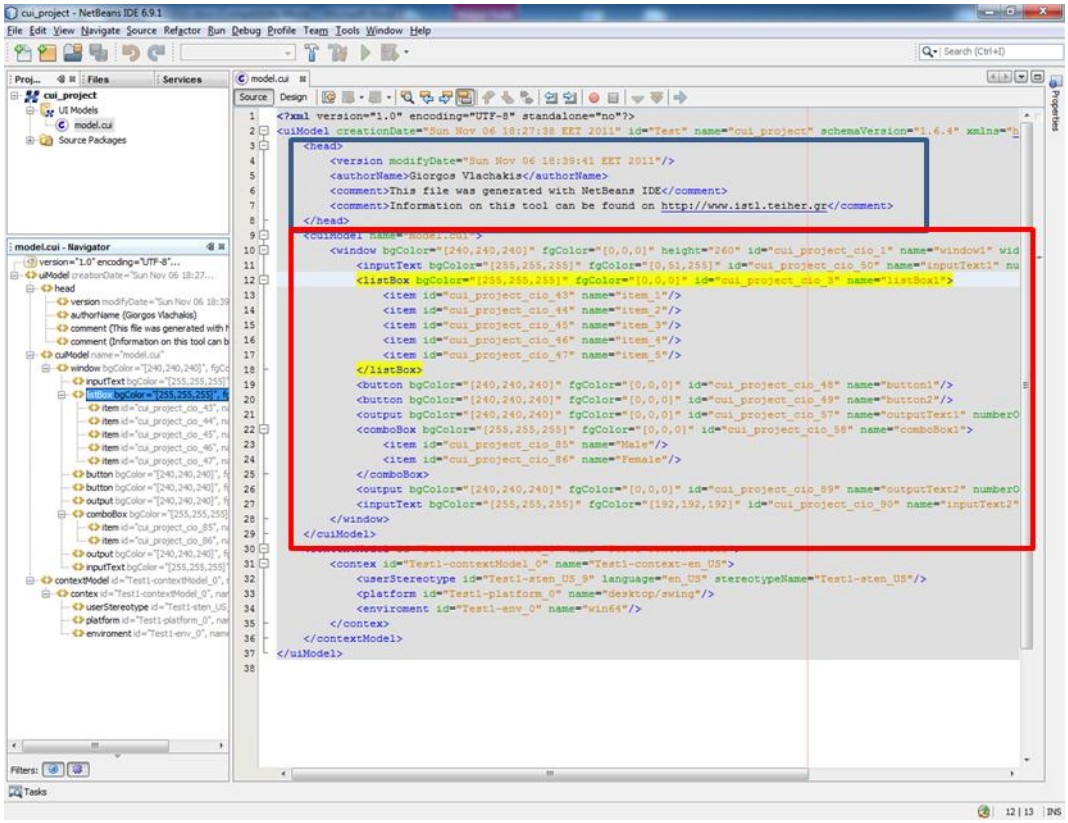
Μετά την ανάλυση της εφαρμογής και των επιμέρους στοιχείων που την απαρτίζουν παρουσιάζονται δύο παραδείγματα χρήσης της εφαρμογής.

Στην Εικόνα 32 παρουσιάζεται μια τυχαία διεπαφή η οποία δημιουργήθηκε με το εργαλείο αποτελείται απο ένα παράθυρο το οποίο περιέχει μία λίστα με πέντε επιλογές, ένα πεδίο για πληκτρολόγηση μεγάλου κειμένου, ένα comboBox με ενδεικτικές επιλογές male, female δύο ετικέτες, ένα πεδίο για εισαγωγή ενός μικρού κειμένου και δυο κουμπιά. Στον Inspector εμφανίζονται μόνο τα κύρια αντικείμενα (κουμπιά, ετικέτες, παράθυρο, λίστα) και όχι τα περιεχόμενα της λίστας και του ComboBox, τα αντικείμενα αυτά φαίνονται ωστόσο στη δεύτερη όψη που περιγράφει τη διεπαφή.



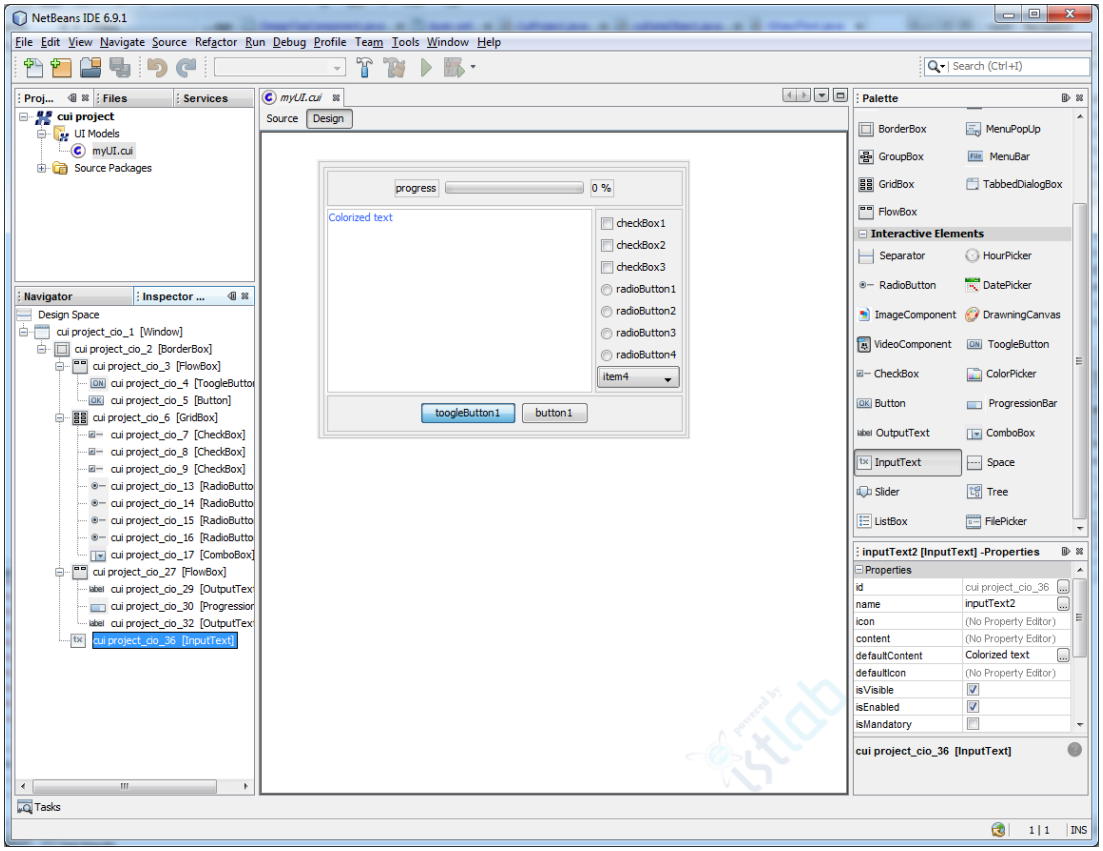
Εικόνα 32: Γραφική απεικόνιση διεπαφής

Η Εικόνα 33 περιέχει το UsiXML κώδικα της διεπαφής που δημιουργήσαμε προηγουμένως. Το XML εκτός απο τα αντικείμενα απο τα οποία αποτελείται η διεπαφή περιέχει και άλλες πληροφορίες όπως το όνομα του πρότζεκτ, το λειτουργικό σύστημα πάνω στο οποίο κτίστηκε, το όνομα του cui αρχείου, το όνομα του δημιουργού και την ημερομηνία δημιουργίας. Στο μπλε πλαίσιο (πεδίο head) βλέπουμε την ημερομηνία δημιουργίας της διεπαφής και κάποια άλλα σχόλια, ενώ στο κόκκινο πλαίσιο (πεδίο cuiModel) γίνεται η περιγραφή της διεπαφής. Τα αντικείμενα παρουσιάζονται ιεραρχικά, ανάλογα με το που περιέχεται το κάθε ένα, και αποθηκεύονται οι ιδιότητες τους, όπως το μοναδικό id έχει το κάθε ένα κι άλλες. Σε αυτή την όψη φαίνονται όλα τα αντικείμενα όπως και τα περιεχόμενα της λίστας τα οποία δεν ήταν ορατά στον inspector.



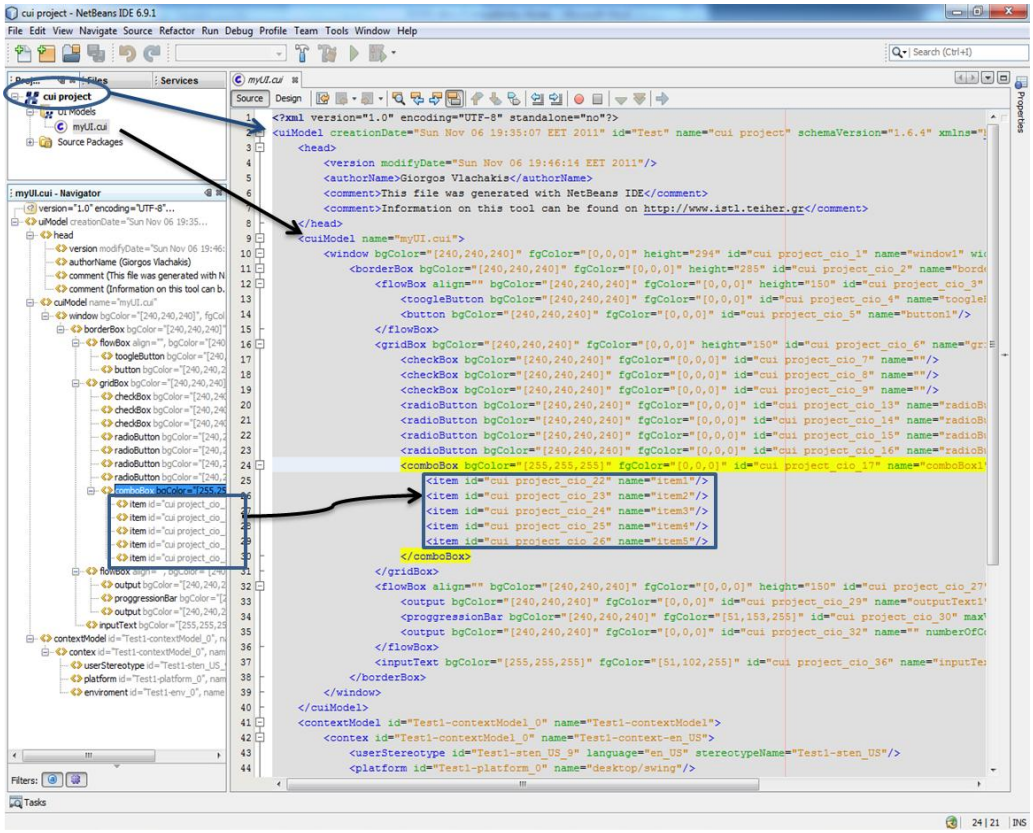
Εικόνα 33: UsiXML Περιγραφή

Στη Εικόνα 34 ομοίως με την Εικόνα 32 έχει δημιουργηθεί μια διεπαφή η οποία αυτή τη φορά είναι πιο σύνθετη. Έχουν χρησιμοποιηθεί επιπλέον αντικείμενα όπως flowbox, borderbox. Τα αντικείμενα αυτά είναι containers οι οποίοι περιέχουν διαχειριστές διάταξης (layout managers) για την οι οποίοι ορίζουν την τοποθέτηση των αντικειμένων. Η χρήση των αντικειμένων με διαχειριστές διάταξης είναι προαιρετική αφού ο χρήστης μπορεί να επιλέξει να σχεδιάσει τη διεπαφή του χωρίς τέτοια αντικείμενα απλά σχεδιάζοντας με τρόπο παρόμοιο με το “Free Design” του Matisse.



Εικόνα 34: Γραφική απεικόνιση διεπαφής

Στην τελευταία εικόνα παρουσιάζεται η UsiXML περιγραφή της προηγούμενης διεπαφής. Στο παράθυρο του navigator έχει επιλεχθεί το combobox και στον editor αυτόματα έχει μαρκαριστεί με κίτρινο χρώμα. Σε μπλε κύκλο επισημαίνεται το project type που δημιουργήθηκε και φαίνεται πως αυτό απεικονίζεται σαν εφαρμογή της πλατφόρμας του NetBeans, ενώ παρουσιάζεται και η λογική δομή των φακέλων οι οποίοι αποτελούν ένα UsiXML project στην πλατφόρμα του NetBeans. Στον φάκελο UI Models βρίσκεται το file type που δημιουργήθηκε (myUI.cui), ενώ το μπλε και το μαύρο βέλος δείχνουν πως καταγράφεται στον κώδικα το project και το αρχείο αντίστοιχα. Τέλος στα δύο πλαίσια φαίνονται τα περιεχόμενα του combobox σε navigator και UsiXML αντίστοιχα.



Εικόνα 35: UsiXML Περιγραφή

6. Επίλογος και Μελλοντικές Εξελίξεις

Η παρούσα εργασία οδήγησε στο πρώτο ολοκληρωμένο περιβάλλον ανάπτυξης για της γλώσσα περιγραφής διεπαφών UsiXML υλοποιώντας όλα τα διαθέσιμα αντικείμενα που το ανάλογο specification ορίζει. Με αυτό τον τρόπο πλέον οι χρήστες μπορούν εύκολα και γρήγορα να σχεδιάσουν με χρήση υψηλών προδιαγραφών εργαλείου διεπαφές για πολλαπλά περιβάλλοντα.

Με την υλοποίηση της εφαρμογής στο περιβάλλον του NetBeans platform και με τη χρησιμοποίηση όλων των APIs που προτείνονται από το NetBeans οποιοσδήποτε επιθυμεί μπορεί να επεκτείνει την εφαρμογή έτσι ώστε να καλύπτει και τα υπόλοιπα επίπεδα αφαίρεσης που αναφέραμε στα προηγούμενα κεφάλαια. Επιπλέον μία άλλου είδους επέκταση θα μπορούσε να είναι αυτή η οποία θα αναλάμβανε τη μετατροπή της σχεδιαζόμενης διεπαφής σε τελική (FUI). Για την επίτευξη αυτού του στόχου, δεδομένης της παρούσης σχεδίασης αλλά και της επιλογής του NetBeans platform ως βάση της σχεδίασης της εφαρμογής μας, δεν θα χρειάζονταν να αλλάξει κάποιος τον υπάρχων κώδικα η την αρχιτεκτονική της ήδη υλοποιημένης εφαρμογής και αυτό πιο συγκεκριμένα χάριν στο modular μοντέλο προγραμματισμού που προωθεί η πλατφόρμα του NetBeans. Έτσι θα μπορούσε να υλοποιηθεί ένα δεύτερο module το οποίο θα ήταν υπεύθυνο γι αυτή την εργασία, το οποίο θα συμπεριλάμβανε το παρών module στις βιβλιοθήκες του, για να μπορέσει να αλληλεπιδρά και να χρησιμοποιεί τη λειτουργικότητα του. Ομοίως το ίδιο θα μπορούσε να συμβεί και για τα υπόλοιπα επίπεδα αφαίρεσης.

7. Βιβλιογραφία

1) Wikipedia: http://wikipedia.org

2) UsiXML: http://www.usixml.org

3) NetBeans: http://netbeans.org

4) Eclipse: http://www.eclipse.org

5) Qt: http://qt.nokia.com

6) Chameleon Reference Framework (CRF): http://www.w3.org/2005/Incubator/model-based-ui/wiki/Cameleon_reference_framework

7) A Review of XML-compliant User Interface Description Languages, Nathalie Soochow and Jean Vanderdonckt

8) A unifying reference framework for multi-target user interfaces http://iihm.imag.fr/publs/2003/Calvary-IwC2003.pdf