Κεφάλαιο 1

Low-Code

Τα πετρογραφικά και οι ζωγραφιές στους τοίχους παλαιών σπηλαίων είναι ένα δείγμα από το πόσο ουσιώδης είναι η οπτική επικοινωνία για τον άνθρωπο. [5]

1.1 Ορισμός

Μια **Πλατφόρμα Ανάπτυξης Εφαρμογών σε Low-Code** (LCAP) είναι μια πλατφόρμα ανάπτυξης λογισμικού που υποστηρίζει την ταχεία ανάπτυξη και διαχείριση εφαρμογών. Συνήθως είναι Platform-as-a-service (PaaS) cloud μοντέλα, και χρησιμοποιείται ελάχιστος ή και μηδενικός δομημένος προγραμματισμός (structured programming).

Για τον προγραμματισμό παρέχεται γραφικό περιβάλλον με οπτικές αφαιρέσεις (visual abstractions), μάλιστα επιτρέποντας χρήστες χωρίς προγραμματιστική εμπειρία να συνεισφέρουν στην ανάπτυξη του λογισμικού χωρίς είναι αναγκαία η βοήθεια των προγραμματιστών. Έτσι οι προγραμματιστές εστιάζουν παραπάνω στη *σχεδίαση* της εφαρμογής, χωρίς να ξοδεύουν χρόνο άσκοπα σε λεπτομέρειες.

Με λίγα λόγια, σκοπός είναι η παραγωγική ανάπτυξη λογισμικού με τη λιγότερη δυνατή προσπάθεια και με χαμηλότερο κόστος, και η εύκολη προσαρμογή του λογισμικού στις ταχέως μεταβαλλόμενες συνθήκες των σημερινών λειτουργικών συστημάτων. Η χρήση των LCAP έχει τύχει θετικής αποδοχής από τη βιομηχανία και η υιοθέτησή τους αυξάνεται συνεχώς. [1, 2, 7]

> "When you can visually create new business applications with minimal hand-coding –when your developers can do more of greater value, faster– that's low-code." [8]

1.2 Ιστορία

Φυσικά η μηχανική λογισμικού έχει περάσει πολλά στάδια στην ιστορία της μέχρι να αρχίσουμε να αναφερόμαστε και σε χρήση αφαιρέσεων. Μέχρι τα μέσα του 1970,

η ανάπτυξη πληροφοριακών συστημάτων έμοιαζε παραπάνω ως μια τέχνη παρά ως επιστήμη, καθώς δεν ακολουθούσε κάποια δόμηση. Αντιθέτως, ο προγραμματιστής λάμβανε μια σειρά από απαιτήσεις και ανάγκες από την πλευρά του χρήστη, και μετά από ένα διάστημα παρέδιδε ένα σύστημα που συνήθως δεν κάλυπτε εξ' ολοκλήρου όλες τις απαιτήσεις του χρήστη αλλά ήταν σίγουρα καλύτερο από το τίποτα. Η συγκεκριμένη μέθοδος, αποκαλούμενη ως **κλασική μέθοδος**, χαρακτηρίζεται από ανεπίσημες οδηγίες, έλλειψη τυποποίησης και αναφορών (documentation).

Η ανάγκη για αύξηση της παραγωγικότητας στον κύκλο ζωής έκδοσης λογισμικού (software release life cycle)[1] οδήγησε στη δημιουργία **επίσημων μεθόδων** στα τέλη της δεκαετίας του 1970. Παράδειγμα αυτών των τεχνικών είναι η χρήση δομημένης ανάλυσης. Έτσι, οι μηχανικοί μπορούσαν να φτιάξουν διαγράμματα ροών δεδομένων (dataflow diagrams), μοντέλα οντοτήτων-συσχετήσεων (ER – entity-relationship models), δημιουργώντας μια συστημική περιγραφή του λογικού και φυσικού μέρους του πληροφοριακού συστήματος που ανέπτυσσαν.

Με την περαιτέρω ανάπτυξη των προσωπικών υπολογιστών στα μέσα της δεκαετίας του 1980, η χρήση επίσημων μεθόδων ήταν μονόδρομος, και μάλιστα οδήγησε στην άνθηση αυτόματων περιβαλλόντων και εργαλείων. Αυτά τα περιβάλλοντα, γνωστά ως CASE, επέτρεπαν τους μηχανικούς να καταγράφουν και να μοντελοποιούν ένα πληροφοριακό σύστημα από τις αρχικές περιγραφές του χρήστη ως τη σχεδίαση και την υλοποίηση και να εκτελούν δοκιμές για τη συνέπειά του. [4]

1.2.1 Computer-Aided Software Enginnering (CASE)

Σε αντίθεση με τα υπόλοιπα εργαλεία προγραμματισμού –που στόχος τους είναι να βοηθήσουν στην επεξεργασία πηγαίου κώδικα–, τα CASE περιβάλλοντα βοηθούν στη *μεθοδολογία* της ανάπτυξης λογισμικού: στην ανάλυση απαιτήσεων, στη σχεδίαση και στον έλεγχο. Είναι το δεξί χέρι ενός μηχανικού λογισμικού, προσφέροντας υποστήριξη για πολλές εργασίες που τον δυσκολεύουν, αυξάνοντας την παραγωγικότητα και την ποιότητα της δουλειάς του. [5]

Τα CASE περιβάλλοντα έθεσαν τα θεμέλια για τη δημιουργία νέων υποδειγμάτων, όπως ο οπτικός προγραμματισμός (visual programming) και ο προγραμματισμός που βασίζεται σε μοντέλα. [3]

1.2.2 Model-driven Architecture

Οι αρχιτεκτονικές που βασίζονται σε μοντέλα (model-driven architecture - MDA) διαχωρίζουν τη λειτουργικότητα μιας εφαρμογής από την υλοποίησή της σε μια συγκεκριμένη πλατφόρμα, προσφέροντας ένα υψηλότερο επίπεδο αφαίρεσης. Στόχος είναι

[1] Πρόκειται μια έννοια που αναφέρεται στις φάσεις ανάπτυξης και ύπαρξης ενός λογισμικού. Ξεκινάει από τη σύλληψη της ιδέας, τη μελέτη για τις απαιτήσεις του, την υλοποίηση του, τη διάθεση του προϊόντος στο πελάτη, τη υποστήριξή του με ενημερώσεις και τέλος την απόσυρσή του.

οι προγραμματιστές να εστιάζουν περισσότερο στον σχεδιασμό και λιγότερο στο να λύνουν θέματα που αφορούν την πλατφόρμα υλοποίησης.

Μέχρι πρότινος η ανάπτυξη λογισμικού αφορούσε καθαρά δομικό κώδικα. Η MDA άλλαξε τον τρόπο σκέψης των προγραμματιστών, καθώς πλέον επικεντρώνονταν παραπάνω στον σωστό διαχωρισμό των χαρακτηριστικών, στην αφαιρετικότητα και στην αυτοματοποίηση. [6, 2]

Βιβλιογραφία

[1] Alexander C. Bock και Ulrich Frank. «Low-Code Platform». Στο: *Business and Information Systems Engineering* 63 (6 Δεκ. 2021), σσ. 733–740. ISSN: 18670202. DOI: 10.1007/s12599-021-00726-8.

[2] Alessio Bucaioni, Antonio Cicchetti και Federico Ciccozzi. «Modelling in low-code development: a multi-vocal systematic review». Στο: *Software and Systems Modeling* 21 (5 Οκτ. 2022), σσ. 1959–1981. ISSN: 16191374. DOI: 10.1007/s10270-021-00964-0.

[3] Albert E Case. *Computer-aided software engineering (case): technology for improving software development productivity*. 1985.

[4] E. J. Chikofsky. *Software Development — Computer-Aided Software Engineering (CASE)*.

[5] D. L. Kuhn. *Selecting and Effectively Using a Computer-Aided Software Engineering Tool*. 1989.

[6] *MDA FAQ | Object Management Group — omg.org*. https://www.omg.org/mda/faq_mda.htm. [Accessed 11-10-2024].

[7] Apurvanand Sahay κ.ά. «Supporting the understanding and comparison of low-code development platforms». Στο: *Proceedings - 46th Euromicro Conference on Software Engineering and Advanced Applications, SEAA 2020*. Institute of Electrical και Electronics Engineers Inc., Αύγ. 2020, σσ. 171–178. ISBN: 9781728195322. DOI: 10.1109/SEAA51224.2020.00036.

[8] *What Is Low-Code? | IBM — ibm.com*. https://www.ibm.com/topics/low-code. [Accessed 11-10-2024].