Προηγμένες Τεχνικές Προγραμματισμού
Sleeping Barber (Version 4)

Δασούλας Ιωάννης – 1053711

Περιγραφή σεναρίου υλοποίησης

Η υλοποίηση αποτελείται από το γραφικό περιβάλλον gui (instance της BarberGui), από την είσοδο (instance της Entrance), από τον χώρο αναμονής (instance της WaitingRoom), από τον χώρο εργασίας (instance της Service Room), από τον κουρέα (instance της Barber) και από τους δέκα πελάτες (instances της Person). Τα active αντικείμενα της εφαρμογής είναι οι πελάτες και ο κουρέας.

Ο κουρέας εργάζεται στον χώρο εργασίας, εξυπηρετώντας έναν πελάτη κάθε φορά. Όταν δεν υπάρχει κάποιος πελάτης που περιμένει να εξυπηρετηθεί, ξεκουράζεται στον χώρο αναμονής. Επίσης, μετά από πέντε συνεχόμενα κουρέματα, κάνει διάλειμμα και μεταβαίνει στον χώρο αναμονής για να ξεκουραστεί. Λόγω του κορονοϊού, στον χώρο αναμονής επιτρέπεται να υπάρχουν μέχρι τέσσερις άνθρωποι (active objects) κάθε χρονική στιγμή. Για αυτόν τον λόγο, στο προτελευταίο του συνεχόμενο κούρεμα πριν την ξεκούρασή του (το τέταρτο συνεχόμενο δηλαδή) ανακοινώνει στους πελάτες ότι για το επόμενο χρονικό διάστημα, το όριο πελατών στον χώρο αναμονής θα είναι τρία, αντί για τέσσερα, ώστε να μπορεί και αυτός να ξεκουραστεί εκεί, τηρώντας τους κανόνες ασφαλείας. Όταν τελειώνει η ανάπαυσή του και επιστρέφει στη θέση εργασίας, ανακοινώνει ξανά στους πελάτες ότι το όριο ατόμων στον χώρο αναμονής επανέρχεται στον αριθμό τέσσερα. Αυτή η διαδικασία επαναλαμβάνεται συνεχώς.

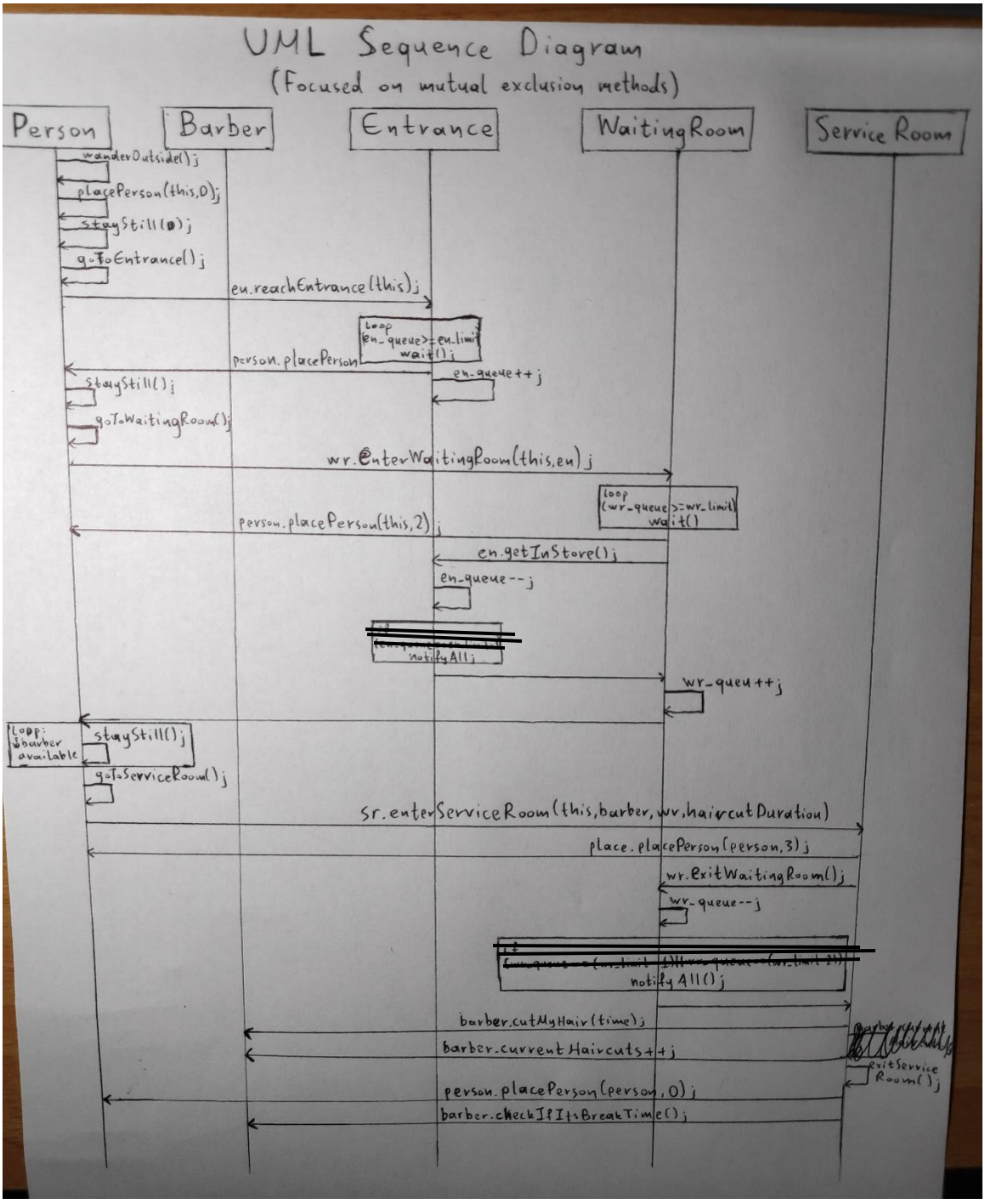
Οι πελάτες κινούνται κυκλικά σε τέσσερις θέσεις (0 έως 3). Η θέση 0 είναι ο χώρος εκτός του καταστήματος στον οποίο οι πελάτες μπορούν να βρίσκονται χωρίς περιορισμούς. Για να εισέλθουν στο κατάστημα, μετακινούνται στη θέση 1, στην είσοδο δηλαδή, στην οποία έχει επίσης τεθεί περιορισμός των ατόμων που μπορούν να στέκονται εκεί, περιμένοντας να προχωρήσουν στον χώρο αναμονής. Ο περιορισμός είναι έως τρία άτομα κάθε στιγμή. Οι πελάτες που στέκονται στην είσοδο προχωρούν ένας-ένας στον χώρο αναμονής (θέση 2), όταν εκεί αδειάσει κάποια θέση. Προχωράει ο πελάτης που συνειδητοποιεί πρώτος ότι άδειασε μία θέση. Αντίστοιχα, οι πελάτες στον χώρο αναμονής περιμένουν είτε ο κουρέας να τελειώσει με τον πελάτη που εξυπηρετεί, είτε να επιστρέψει στην θέση εργασίας του μετά από το διάλειμμά του. Όταν ένας πελάτης καταλάβει μία τέτοια αλλαγή, εισέρχεται στην αίθουσα εργασίας, δηλαδή στη θέση 3, κουρεύεται και αποχωρεί από το κατάστημα (θέση 0). Έπειτα, επαναλαμβάνει τον κύκλο αυτό.

Προηγμένες Τεχνικές Προγραμματισμού
Sleeping Barber (Version 4)

Δασούλας Ιωάννης – 1053711

Sequence Diagram αμοιβαίου αποκλεισμού

Ο τρόπος με τον οποίο τα νήματα συγχρονίζονται φαίνεται στο παρακάτω sequence diagram, επικεντρωμένο στις monitor κλάσεις.



Προηγμένες Τεχνικές Προγραμματισμού
Sleeping Barber (Version 4)

Δασούλας Ιωάννης – 1053711

Η κλάση Entrance για την θέση 1, η WaitingRoom για την θέση 2 και η ServiceRoom για την θέση 3, υλοποιούν τον αμοιβαίο αποκλεισμό με monitor, η κάθε μία για την αντίστοιχη θέση.

Όταν ένας πελάτης θέλει να μεταβεί στην είσοδο στέλνει μήνυμα en.reachEntrance(this). Όσο η ουρά στην είσοδο είναι γεμάτη, εκτελεί wait(), αλλιώς τοποθετείται στην ουρά με το μήνυμα person.placePerson(person,1) και η ουρά αυξάνεται κατά ένα.

Μένει ακίνητος για λίγο (για καλύτερη απεικόνιση στο gui) και έπειτα ζητάει να προχωρήσει στον χώρο αναμονής, στέλνοντας μήνυμα enterWaitinRoom(this, 2). Όσο η ουρά στην αίθουσα αναμονής είναι γεμάτη, εκτελεί wait(), αλλιώς τοποθετείται στην αίθουσα αναμονής με μήνυμα person.placePerson (this.2) στην κλάση Person και στέλνει μήνυμα en.getInStore() στην κλάση Entrance για να γνωστοποιήσει ότι αφήνει την είσοδο για να προχωρήσει. Η ουρά της εισόδου μειώνεται κατά 1 και γίνεται notifyAll() για να μπορεί να προχωρήσει κάποιος από τους πελάτες που βρίσκεται σε wait(). Η ουρά της αίθουσας αναμονής αυξάνεται κατά ένα.

Έπειτα, είναι η ώρα ο πελάτης να προχωρήσει για να εξυπηρετηθεί. Σε ένα while βρόγχο ελέγχει, αρχικά, αν ο κουρέας εργάζεται εκείνη τη στιγμή καθώς μπορεί να κάνει το διάλειμμά του. Αν δεν εργάζεται, ο πελάτης “κοιμάται” για λίγο, εκτελώντας τη μέθοδο sleep(), περιμένοντας να επιστρέψει ο κουρέας στα καθήκοντά του. Όταν συνειδητοποιήσει ότι ο κουρέας εργάζεται ξανά, στέλνει μήνυμα sr.enterServiceRoom(this,barber,wr,haircutDuration) στην κλάση ServiceRoom για να ζητήσει να εισέλθει στον χώρο εξυπηρέτησης. Η μέθοδος αυτή είναι synchronized κι έτσι ένας μόνο πελάτης εξυπηρετείται κάθε φορά. Ο πελάτης τοποθετείται στη θέση εξυπηρέτησης με το μήνυμα person.placePerson(person,3) και γνωστοποιείται η αποχώρηση από τον χώρο αναμονής με το μήνυμα wr.exitWaitingRoom() στην κλάση WaitingRoom. Με το μήνυμα αυτό, η ουρά του χώρου αναμονής μειώνεται κατά ένα και εκτελείται notifyAll() για να μπορεί κάποιος πελάτης, που είναι σε wait() στην είσοδο, να προχωρήσει στον χώρο αναμονής. Ο πελάτης στη συνέχεια κουρεύεται για χρόνο που προσδιορίζεται στην δημιουργία του (haircutDuration) και στέλνεται μήνυμα cutMyHair() στον barber. Αυξάνεται κατά ένα ο αριθμός συνεχόμενων κουρεμάτων του κουρέα, αποχωρεί ο πελάτης από το κατάστημα με μήνυμα person.placePerson (person,0) και ο κουρέας ελέγχει αν έχει κουρέψει αρκετούς ανθρώπους ώστε να κάνει για λίγο διάλειμμα με το μήνυμα barber.checkIfItsBreakTime(). Αν ο κουρέας καταλάβει ότι το επόμενο κούρεμα θα είναι το τελευταίο πριν το διάλειμμά του, ενημερώσει τους πελάτες στην είσοδο ότι το όριο πελατών στον χώρο αναμονής μειώνεται σε τρία άτομα για ένα χρονικό διάστημα, ώστε να μην εισέλθει τέταρτος την ώρα που θέλει να ξεκουραστεί και αναγκαστεί να τον διώξει, κι έτσι να τηρεί τους κανόνες ασφαλείας.

Προηγμένες Τεχνικές Προγραμματισμού
Sleeping Barber (Version 4)

Δασούλας Ιωάννης – 1053711

Προβλήματα κατά την υλοποίηση και λύσεις

Ο σωστός συγχρονισμός των νημάτων συχνά επιφυλάσσει δυσκολίες. Ένα από τα πρώτα προβλήματα που συναντήθηκε ήταν η κατανόηση της σωστής διαχείρισης των κοινών πόρων καθώς στο συγκεκριμένο πρόβλημα διακρίνονται τρεις κοινοί πόροι, τρεις θέσεις για τις οποίες η κάθε μία χρειάζεται ξεχωριστή διαχείριση. Ένα άλλο πρόβλημα είναι ότι τα δωμάτια που απαιτούν αμοιβαίο αποκλεισμό είναι διαδοχικά μεταξύ τους, οπότε πρέπει να υπάρχει κατάλληλη συνεννόηση των monitor κλάσεων. Αυτό επιλύθηκε χωρίζοντας τις monitor κλάσεις σε δύο μεθόδους, μία για είσοδο και μία για έξοδο από το κάθε δωμάτιο. Η μέθοδος που εισάγει έναν πελάτη σε ένα δωμάτιο ενημερώνει την μέθοδο της αντίστοιχης κλάσης που είναι υπεύθυνη για το προηγούμενο δωμάτιο ότι αποχώρησε από αυτό. Με την συνεχή δηλαδή επικοινωνία των κοινών πόρων λύθηκε μεγάλο κομμάτι του προβλήματος.

Μία άλλη δυσκολία ήταν η σωστή λειτουργία του κουρέα σε σχέση με τους πελάτες. Το πρώτο πρόβλημα ήταν η απαγόρευση πελατών να εισέρχονται στο δωμάτιο εξυπηρέτησης ενώ κάνει διάλειμμα, κάτι που επιτεύχθηκε με μία Boolean μεταβλητή και έλεγχο του πελάτη για την τιμή της. Το πιο σημαντικό, όμως πρόβλημα πιθανόν είναι η αλλαγή του ορίου των πελατών στην αίθουσα αναμονής κατά την εκτέλεση. Για το συγκεκριμένο πρόβλημα, αυτό λύθηκε μειώνοντας το όριο της αίθουσας αναμονής στο προτελευταίο κούρεμα, πριν την ανάπαυση, του κουρέα, ώστε να μην εισέλθει τέταρτος στο τελευταίο του κούρεμα και αναγκαστεί να τον διώξει. Μια πιο modular λύση είναι να διώχνει τον/τους επιπλέον πελάτες όταν θέλει να ξεκουραστεί, έχοντας δημιουργήσει πρώτα μία λίστα αποθήκευσης των πελατών που περιμένουν, όμως αυτό δεν συνάδει με την λογική ενός καταστήματος.

Η σωστή τμηματοποίηση και διαχωρισμός των λειτουργιών και του απαραίτητου κώδικα ήταν άλλο ένα πρόβλημα, για το οποίο έγινε αρκετή προσπάθεια ώστε να λυθεί όσο το δυνατό περισσότερο και να είναι φανερή η λειτουργία του κώδικα κάθε φορά.

Προηγμένες Τεχνικές Προγραμματισμού
Sleeping Barber (Version 4)

Δασούλας Ιωάννης – 1053711

Σχόλια/Παρατηρήσεις

Το πρόβλημα του κοιμώμενου κουρέα είναι πολύ ενδιαφέρον στην μελέτη της λειτουργίας των νημάτων. Ιδιαίτερα στην έκδοση 4, περιέχει διάφορα σενάρια συντονισμού των νημάτων που ζητούν λύση και αποτελεί καλή εξάσκηση για αυτό.

Εναλλακτικές υλοποιήσεις

Ένας πιο σύντομος τρόπος που θα μπορούσε να υλοποιηθεί ο συγχρονισμός, είναι με την χρήση σεμαφόρων. Οι τρεις ελεγκτές αντικαθιστούνται από τρία instances της κλάσης Semaphore, καθένα από τα οποία δημιουργείται με αρχική τιμή ίση με το όριο του δωματίου για το οποίο φτιάχτηκε. Όταν ένας πελάτης επιθυμεί να εισέλθει στο αντίστοιχο δωμάτιο εκτελεί acquire(), και όταν αποχωρεί, εκτελεί release(). Επίσης, χρειάζονται και δύο σεμαφόροι για συγχρονισμό των πελατών ανάμεσα στις θέσεις 1-2 και 2-3 αντίστοιχα. Παρόλα αυτά, η υλοποίηση με σεμαφόρους, λόγω της μεγαλύτερης ευκολίας της, δεν έχει το ίδιο ενδιαφέρον, από προγραμματιστική άποψη.