Projet IS  
LALLEMENT – RAMUZAT – RIGAL

## Problème initial

Satisfaction trop faible des clients car :

* Manque de place dans les ateliers
* Files d’attentes trop longues
* Ateliers trop durs

## Solutions envisageables :

Réduction du nombre de clients (en dernier ressort)

Zone supplémentaire devient atelier (sauf filiformes)

## Impératifs :

Simulation réutilisable dans contextes différents (chgt de scénarios simple)

Estimation de la satisfaction par mois de l’année

* Ratio efficacité réelle / théorique
* Temps attente des curistes par jour

## Libertés prises :

* Suppression des médecins -> créations des cures le dimanche à la création des patients sans classe médecin

# Etablissement

## Caractéristiques

Max 180 curistes/jour

Ouverture Mars à Septembre / Lundi à Samedi / 07h à 14h

8 zones d’ateliers et 1 non utilisée

1 zone de détente

Distances entre ateliers

## Fonctionnement

//

## Modélisation

Entité

Classe SpaResort

int maxClients

time openingHour

time closingHour

int[] timetable

Treatment[] treatments

int[] distances

# Ateliers

## Caractéristiques

Ouverts de 7h15 à 14h00 sauf Filiforme 10h à 13h

Heure fixe ou horaires libres

Durée fixe

Files d’attentes organisées ou non organisées avec places max

Soumis à des arrêts (Matériel ou personnels malades / Lois à déterminer)

Responsable d’atelier

## Fonctionnement

Ateliers en cours lors de la fermeture se finissent (i.e. vont jusqu’au bout ou s’arrêtent immédiatement ??) et curistes renoncent à ateliers non entamés

Chaque atelier a une efficacité représentée par des points (curiste gagne % points proportionnels au temps passé dedans)

Heure fixe -> appel (on ne peut pas piquer la place d’un autre)

Responsable pas obligatoirement dans sa zone (équiprobable sa zone ou autre zone)

Responsable forcément 2 minutes consécutives dans sa zone par créneaux d’atelier

## Modélisation

Classe Treatment

time startHour

time endHour

Duration duration

int maxPoint

boolean isFixedHours

boolean isOrganizedWaiting

Patient[] waitingQueue

int maxWaitingPatients

Manager manager

## Questions

Création d’une classe Manager pour gérer le responsable ou pas ?

# Cure

## Caractéristiques

Une cure dure 3 semaines consécutives, un client a 3 cures (pour 1 par année)

Début lundi matin

Min 3 / Max 6 ateliers/jours

## Fonctionnement

Ateliers décidés par médecin le week-end précédant la cure

Mêmes ateliers chaque jour

Deux types de bains à jets (modernes / anciens) -> un seul type affecté par cure

Chaque atelier a une efficacité représentée par des points (curiste gagne % points proportionnels au temps passé dedans)

## Modélisation

Classe Cure

time startDate

int nbrTreatmentsDay

Treatment[] dueTreatments

int maxPoints

int currentPoints

## Questions

Modélisation de chaque semaine par une instance de Cure ? Permet d’accéder aux stats de chacune indépendamment.

# Patients

## Caractéristiques

Peut-être peu scrupuleux

## Fonctionnement

Si plus de place, curiste cherche un autre atelier ou va en zone de détente

Possibilité de replanifier un créneau raté avec responsable

Curiste ne connaît pas les disponibilités des autres ateliers

Curiste peu oublier son créneau à heure fixe quand dans la zone de détente, il cherche alors des trous

Curiste peut rester trop longtemps dans un atelier libre

Curistes peu scrupuleux peut voler la place d’un autre (i.e. « gruger » dans la file) -> + fréquent si file non organisée

## Modélisation

Entité

Classe Patient

boolean isFair

boolean isInRestZone

Duration timeWaited

time startWaiting

time startTreatment

Cure[] curesToDo

Treatment[] missedTreatments

Manager findManager(Treatment managerZone)

void findFreeTreatment(Treatment nextTreatment) (regarde la queue de patients si vide ok)

## Questions

Replanification de créneau raté -> Seulement dans la même journée ?

# Manager

## Caractéristiques

Responsable d’un Atelier

Pas obligatoirement dans sa zone (équiprobable sa zone ou autre zone)

## Fonctionnement

Forcément 2 minutes consécutives dans sa zone par créneaux d’atelier

Possibilité de replanifier un créneau raté par un client

## Modélisation

Entité

Classe Manager

Treatment treatmentResponsivity

boolean isInGoodZone

boolean planMissedTreatment(Patient patient, Treatment missedTreatment)