CAS ZERNE

Présentation du contexte

Rattachée au ministère de l'Intérieur, la Direction de la Sécurité civile (DSC) est la structure centrale responsable de la gestion des risques en France pour les accidents de la vie courante ou les catastrophes majeures. Parmi ses compétences figure la gestion des services départementaux d'incendie et de secours (SDIS).

Dans chaque département, la lutte contre les incendies, le secours aux personnes et la protection des biens et de l'environnement sont pris en charge par le SDIS qui regroupe différentes unités dont :

- le CTA : centre de traitement de l'alerte qui reçoit les appels au secours et déclenche l'alerte,
- le CODIS : centre opérationnel départemental d'incendie et de secours qui gère l'intervention (coordination, demande de renfort…),
- les CIS : centres d'incendie et de secours appelés familièrement "casernes".

Le département du Finistère, par exemple, regroupe 350 sapeurs-pompiers professionnels et plus de 2 000 sapeurs-pompiers volontaires sur 70 CIS. Chaque année, ce département gère environ 50 000 interventions.

Un véhicule de secours doit arriver en moins de 20 minutes sur les lieux de l'intervention n'importe où dans le département, suivant la directive du schéma départemental d'analyse et de couverture des risques.

Pour ce faire, le départ du véhicule doit avoir lieu dans les 7 minutes qui suivent la réception de l'alerte.

Afin de répondre au mieux à cette contrainte, le SDIS 29 modernise son fonctionnement et notamment les moyens de transmission de l'alerte. C'est une évolution extrêmement importante qui implique de nouveaux moyens matériels en informatique et la refonte du système d'information.

Afin de planifier l'activité des pompiers volontaires, tous les dimanches, le chef du centre de secours établit le calendrier des gardes pour la semaine suivante.

Chaque journée est découpée en 4 tranches horaires :

1. nuit : de 0 à 6 heures,
2. matinée : de 6 à 12 heures,
3. après-midi : de 12 à 18 heures,
4. soirée : de 18 à 0 heures le lendemain.

Chaque pompier volontaire indique au chef de centre sa disponibilité pour chaque journée et chaque tranche horaire de la semaine suivante :

1. indisponible : le pompier ne peut pas être sollicité pour cette période de garde,
2. au travail : le pompier travaille pour son employeur, il peut être sollicité pour cette période de garde, même s'il est préférable de l'éviter,
3. disponible : le pompier est libre de tout engagement, il peut donc être sollicité pour cette période de garde.

À partir de ces informations, le chef de centre établit la feuille de garde présentée en **annexe A**. Ce document indique la liste des pompiers volontaires qui devront être prêts à intervenir pour chaque période de garde. Une période de garde est définie par une date et une tranche horaire.

Dans le but d'informatiser la constitution des feuilles de garde et la gestion des alertes, la base de données suivante a été constituée.

VOLONTAIRE (id, nom, prenom, numeroBip, nbGardes)
// Les pompiers volontaires du centre de secours
Clé primaire : id (de type entier)
"nbGardes" : nombre de gardes assurées depuis le début de l'année civile

TRANCHE (id, libelle)
// Les quatre tranches horaires
Clé primaire : id

PERIODE_GARDE (idTranche, datePeriode, nbPompiers)
// Exemple : La 2ème tranche, de 6 heures à 12 heures (matinée), en date du 18/03/2011
Clé primaire : idTranche, datePeriode
Clé étrangère : idTranche en référence à TRANCHE (id)
"nbPompiers" : nombre de volontaires nécessaires pour cette période de garde

DISPONIBILITE (id, libelle)
// 3 lignes : "1, Indisponible", "2, au travail" et "3, disponible"
Clé primaire : id

AVOIR_ACTIVITE (idVolontaire, idTranche, datePeriode, idDisponibilite, deGarde)
// Indique la disponibilité d'un volontaire pour une période de garde
Clé primaire : idVolontaire, idTranche, datePeriode
"deGarde" : 1 si le volontaire a été désigné pour être de garde pour cette période, 0 sinon

Remarque : Les clés étrangères de la table AVOIR_ACTIVITE sont à définir.

Lorsque le chef de centre a désigné les volontaires assurant chaque période de garde, la solution informatique doit permettre d'appeler rapidement une équipe lors d'un départ pour une intervention. Cette partie de l'application doit être réalisée à l'aide d'un langage orienté objet. Un extrait du diagramme de classes utilisé est présenté en **annexe B**, la description littérale des classes en **annexe C**. L'ensemble des objets est instancié à partir de la base de données dès le lancement de l'application.

Le principe est de constituer l'équipe d'intervention en mobilisant en priorité les pompiers disponibles, puis, à défaut, les pompiers actuellement au travail.

La classe "Caserne" est chargée de constituer l'équipe d'intervention et d'appeler chaque pompier mobilisé sur son récepteur d'appel (bip).

La classe "Période" maintient trois collections de pompiers :

- "enMission" : les pompiers déjà mobilisés sur une intervention.
- "auTravail" : les pompiers actuellement chez leur employeur.
- "disponible" : les pompiers disponibles pour une intervention.

La classe technique "Collection" est présentée en **annexe D**.

Travail à faire	
1	Écrire la méthode "GetNuméroBip" de la classe "Pompier".
2	Écrire la méthode "GetStatut" de la classe "Pompier".
3	Écrire le constructeur de la classe "Période".
4	Écrire la méthode "Missionner" de la classe "Période".
5	Écrire la méthode "SelectEquipe" de la classe "Période".
6	Écrire la méthode "AppelEquipe" de la classe "Caserne".

ANNEXE A - Feuille de garde

CIS Ouessant : feuille de garde semaine 23																											
		Lun 06/06				Mar 07/06				Mer 08/06				Jeu 09/06				Ven 10/06				Sam 11/06				Dim 12/06			
VOLONTAIRE	**N° bip**	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
Durand François	2		X	X		X	X								X	X		X	X		X		X		X		X		X
Legall Yasmina	3	X			X	X		X	X	X	X		X		X		X			X	X	X			X	X	X	X	
Dubois Yves	9																												
Martin Alain	17	X	X	X			X	X	X			X	X	X			X	X			X	X		X	X	X		X	X
Dupond Carole	4			X	X	X	X	X		X	X			X	X	X		X	X										
Yayaoui Pierre	11			X	X		X		X	X		X	X	X			X		X	X		X			X	X			X
Carette Patrick	12	X	X																X	X	X	X	X		X	X	X	X	
Fernandez Henri	18	X		X	X		X	X		X		X	X		X	X	X				X	X		X		X			X
Cabon Yohann	9					X		X	X		X	X			X	X		X	X				X	X	X		X	X	
Breton Joëlle	5		X			X				X	X			X	X	X		X		X	X		X					X	X
Dujardin Alex	10	X		X		X	X	X	X		X		X	X	X									X	X	X	X		
Clébert Marc	19	X	X		X						X	X					X	X	X	X	X		X				X	X	X

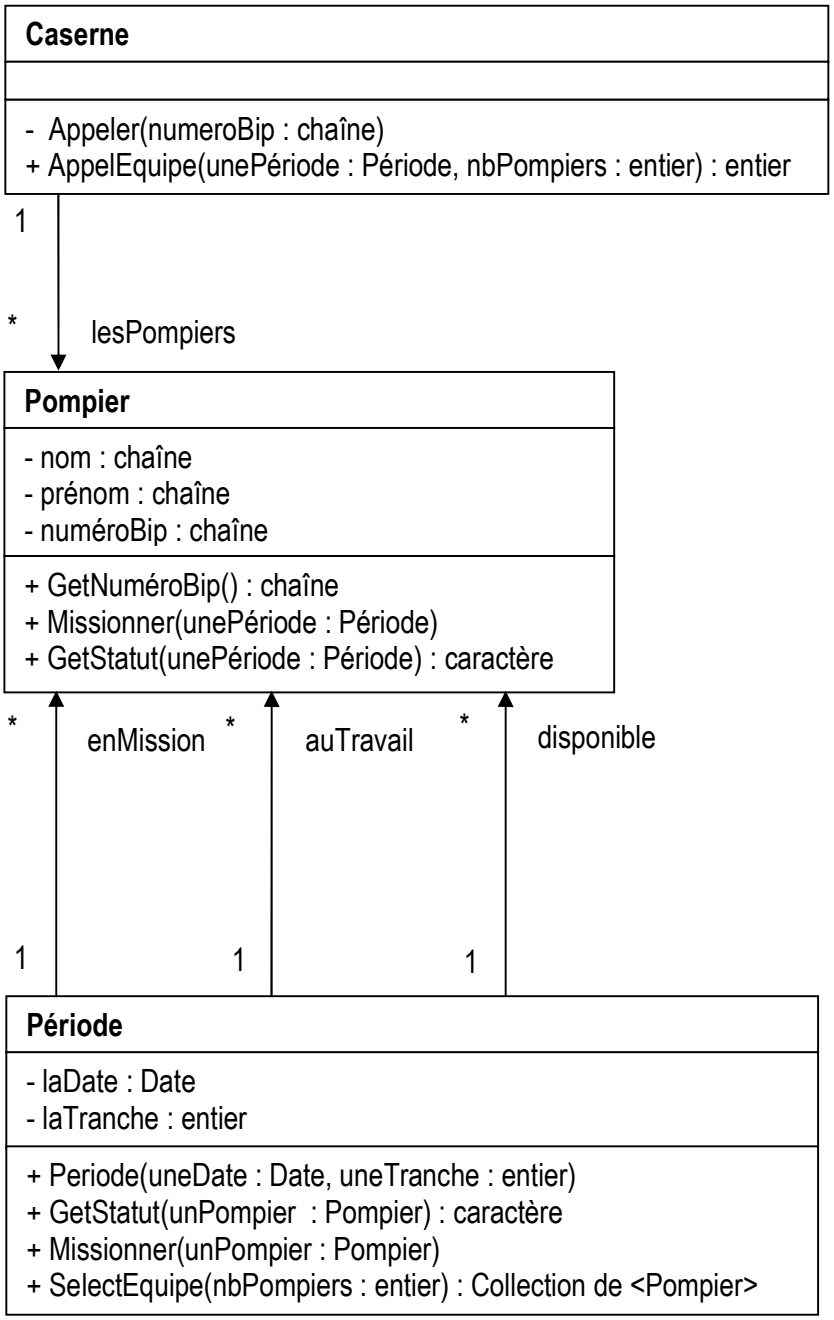
Remarques :

- Chaque case correspond à une tranche horaire d'un jour de la semaine.
- Une case blanche signifie que le volontaire est disponible pour cette tranche horaire.
- Une case hachurée signifie que le volontaire est au travail pendant cette tranche horaire.
- Une case pleine signifie que le volontaire est indisponible pendant cette tranche horaire.
- Une croix indique que le volontaire a été désigné de garde pour cette tranche horaire.

Exemples pour le lundi 06/06 :

- Durand François est disponible pour les quatre tranches horaires. Il est de garde pour les tranches 2 et 3.
- Legall Yasmina est au travail pour les tranches horaires 2, 3 et 4. Elle est de garde pour les tranches 1 et 4.
- Dubois Yves est indisponible pour les quatre tranches horaires, il n'est donc pas de garde pour cette journée.

ANNEXE B - Extrait du diagramme de classes



ANNEXE C - Extrait de la description littérale des classes

Classe Caserne

Privé

lesPompiers : Collection de <Pompier>
// L'ensemble des pompiers de garde pour la caserne
Procédure Appeler(numéroBip : chaîne)
// Appelle automatiquement le numéro de bip passé en paramètre

Public

Fonction AppelEquipe(unePériode : Période, nbPompiers : entier) : entier
// Appelle une équipe de pompiers pour une intervention concernant la période passée en
// paramètre. Le paramètre "nbPompiers" représente le nombre de pompiers à solliciter.
// Si un nombre suffisant de pompiers ne peut pas être sollicité, une équipe incomplète sera
// appelée. Cette fonction modifie le statut des pompiers sélectionnés, les appelle sur leur bip et retourne
// le nombre de pompiers appelés.

Fin Classe

Classe Période

Privé

laDate : Date *// date correspondant à la période de garde*
laTranche : entier *// tranche de la période (1, 2, 3 ou 4)*
enMission : Collection de <Pompier> *// pompiers déjà en intervention*
auTravail : Collection de <Pompier> *// pompiers au travail chez leur employeur*
disponible : Collection de <Pompier> *// pompiers disponibles à leur domicile*

Public

Constructeur Période(uneDate : Date, uneTranche : entier)
// Ce constructeur valorise les attributs privés de l'objet Période

Fonction GetStatut(unPompier : Pompier) : caractère
// Retourne 'm' si le pompier "unPompier" est en mission pour la période courante
// Retourne 't' si le pompier "unPompier" est au travail pour la période courante
// Retourne 'd' si le pompier "unPompier" est disponible pour la période courante

Procédure Missionner(unPompier : Pompier)
// Modifie le statut du pompier passé en paramètre : de "disponible" à "enMission"
// ou de "auTravail" à "enMission".

Fonction SelectEquipe(nbPompiers : entier) : Collection de <Pompier>
// Retourne une collection d'au maximum "nbPompiers" pompiers pouvant être mobilisés pour
// une intervention lors de la période courante. Si un nombre suffisant de pompiers ne peut pas
// être sélectionné, une équipe incomplète sera constituée.
// Cette fonction mobilise en priorité les pompiers disponibles, puis ceux qui
// sont au travail si nécessaire. Le statut des pompiers n'est pas modifié par cette méthode.

Fin Classe

Classe Pompier
Privé
nom : chaîne // *nom du pompier*
prénom : chaîne // *prénom du pompier*
numéroBip : chaîne // *numéro de bip du pompier*
Public
Fonction GetNuméroBip() : chaîne
// retourne le numéro de bip du pompier

Fonction GetStatut(unePériode : Période) : caractère
// Retourne le statut du pompier courant pour la période passée en paramètre :
// 'm' pour en mission, 't' pour au travail, ou 'd' pour disponible

Procédure Missionner(unePériode : Période)
// Modifie le statut du pompier courant pour la période passée en paramètre
Début
unePériode.Missionner(this)
fin
Fin Classe

ANNEXE D - Extrait de la classe Collection

Classe Collection de <TypeElément> *// où TypeElément peut être un type simple ou une classe*
Public
Fonction Cardinal() : entier
// nombre d'éléments de la collection
Fonction Contient(unObjet : TypeElément) : booléen
// renvoie vrai si "unObjet" appartient à la collection
Fonction Index(unObjet : TypeElément) : entier
// renvoie l'index d'un objet de la collection, le premier objet de la collection a pour index 0
Fonction Extraire(unIndex : entier) : TypeElément
// retourne l'objet d'index "unIndex"
Procédure Ajouter(unObjet : TypeElément)
// ajoute l'objet "unObjet" à la collection
Procédure Enlever(unObjet : TypeElément)
// supprime l' objet "unObjet" de la collection
Fin Classe

Exemple d'utilisation d'une collection :

```
uneCol : Collection de <Date>
uneDate : Date
uneCol = new Collection de <Date>()
uneCol.Ajouter(uneDate)
Pour chaque uneDate dans uneCol faire
    // Traitement de la Date "uneDate"
Fin Pour
```