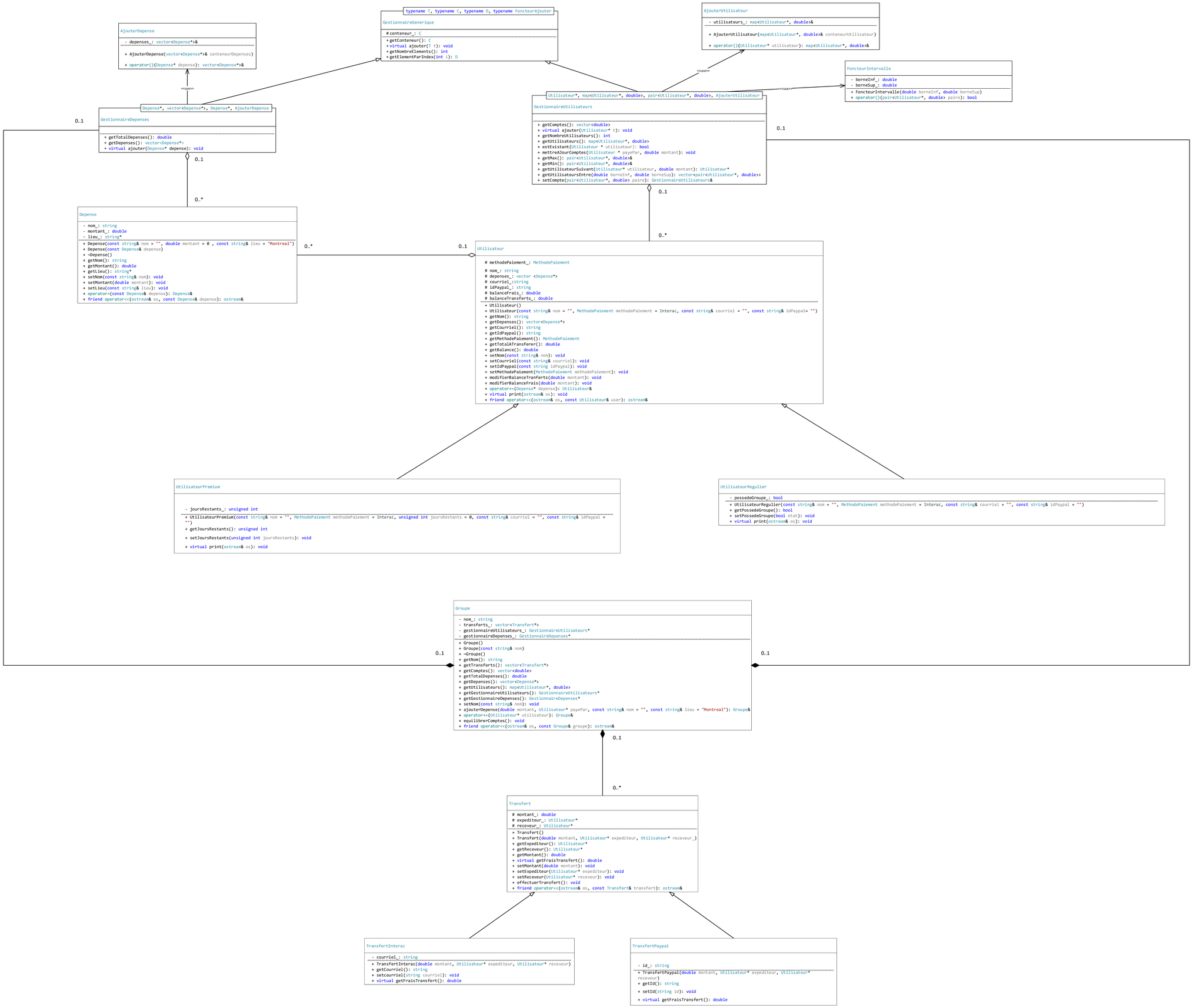
AjouterDepense
- depenses_: vector<Depense*>&
+ AjouterDepense(vector<Depense*>& conteneurDepenses)
+ operator()(Depense* depense): vector<Depense*>&
typename T, typename C, typename D, typename FoncteurAjouter
GestionnaireGenerique
conteneur_: C
+ getConteneur(): C
+ virtual ajouter(T t): void
+ getNombreElements(): int
+ getElementParIndex(int i): D
AjouterUtilisateur
- utilisateurs_: map<Utilisateur*, double>&
+ AjouterUtilisateur(map<Utilisateur*, double>& conteneurUtilisateur)
+ operator()(Utilisateur* utilisateur): map<Utilisateur*, double>&
<<use>>
FoncteurIntervalle
- borneInf_: double
- borneSup_: double
+ FoncteurIntervalle(double borneInf, double borneSup)
+ operator()(pair<Utilisateur*, double> paire): bool
<<use>>
<<use>>
Depense*, vector<Depense*>, Depense*, AjouterDepense
GestionnaireDepenses
+ getTotalDepenses(): double
+ getDepenses(): vector<Depense*>
+ virtual ajouter(Depense* depense): void
Utilisateur*, map<Utilisateur*, double>, pair<Utilisateur*, double>, AjouterUtilisateur
GestionnaireUtilisateurs
+ getComptes(): vector<double>
+ virtual ajouter(Utilisateur* t): void
+ getNombreUtilisateurs(): int
+ getUtilisateurs(): map<Utilisateur*, double>
+ estExistant(Utilisateur * utilisateur): bool
+ mettreAJourComptes(Utilisateur * payePar, double montant): void
+ getMax(): pair<Utilisateur*, double>&
+ getMin(): pair<Utilisateur*, double>&
+ getUtilisateurSuivant(Utilisateur* utilisateur, double montant): Utilisateur*
+ getUtilisateursEntre(double borneInf, double borneSup): vector<pair<Utilisateur*, double>>
+ setCompte(pair<Utilisateur*, double> paire): GestionnaireUtilisateurs&
0..1
0..1
0..1
0..*
0..1
0..*
Depense
- nom_: string
- montant_: double
- lieu_: string*
+ Depense(const string& nom = "", double montant = 0 , const string& lieu = "Montreal")
+ Depense(const Depense& depense)
+ ~Depense()
+ getNom(): string
+ getMontant(): double
+ getLieu(): string*
+ setNom(const string& nom): void
+ setMontant(double montant): void
+ setLieu(const string& lieu): void
+ operator=(const Depense& depense): Depense&
+ friend operator<<(ostream& os, const Depense& depense): ostream&
0..*
0..1
Utilisateur
methodePaiement_: MethodePaiement
nom_: string
depenses_: vector <Depense*>
courriel_:string
idPaypal_: string
balanceFrais_: double
balanceTransferts_: double
+ Utilisateur()
+ Utilisateur(const string& nom = "", MethodePaiement methodePaiement = Interac, const string& courriel = "", const string& idPaypal= "")
+ getNom(): string
+ getDepenses(): vector<Depense*>
+ getCourriel(): string
+ getIdPaypal(): string
+ getMethodePaiement(): MethodePaiement
+ getTotalATransferer(): double
+ getBalance(): double
+ setNom(const string& nom): void
+ setCourriel(const string& courriel): void
+ setIdPaypal(const string idPaypal): void
+ setMethodePaiement(MethodePaiement methodePaiement): void
+ modifierBalanceTranferts(double montant): void
+ modifierBalanceFrais(double montant): void
+ operator++(Depense* depense): Utilisateur&
+ virtual print(ostream& os): void
+ friend operator<<(ostream& os, const Utilisateur& user): ostream&
UtilisateurPremium
- joursRestants_: unsigned int
+ UtilisateurPremium(const string& nom = "", MethodePaiement methodePaiement = Interac, unsigned int joursRestants = 0, const string& courriel = "", const string& idPaypal = "")
+ getJoursRestants(): unsigned int
+ setJoursRestants(unsigned int joursRestants): void
+ virtual print(ostream& os): void
UtilisateurRegulier
- possedeGroupe_: bool
+ UtilisateurRegulier(const string& nom = "", MethodePaiement methodePaiement = Interac, const string& courriel = "", const string& idPaypal = "")
+ getPosseDeGroupe(): bool
+ setPosseDeGroupe(bool etat): void
+ virtual print(ostream& os): void
0..1
0..1
Groupe
- nom_: string
- transferts_: vector<Transfert*>
- gestionnaireUtilisateurs_: GestionnaireUtilisateurs*
- gestionnaireDepenses_: GestionnaireDepenses*
+ Groupe()
+ Groupe(const string& nom)
+ ~Groupe()
+ getNom(): string
+ getTransferts(): vector<Transfert*>
+ getComptes(): vector<double>
+ getTotalDepenses(): double
+ getDepenses(): vector<Depense*>
+ getUtilisateurs(): map<Utilisateur*, double>
+ getGestionnaireUtilisateurs(): GestionnaireUtilisateurs*
+ getGestionnaireDepenses(): GestionnaireDepenses*
+ setNom(const string& nom): void
+ ajouterDepense(double montant, Utilisateur* payePar, const string& nom = "", const string& lieu = "Montreal"): Groupe&
+ operator+=(Utilisateur* utilisateur): Groupe&
+ equilibrerComptes(): void
+ friend operator<<(ostream& os, const Groupe& groupe): ostream&
0..1
0..*
Transfert
montant_: double
expediteur_: Utilisateur*
receveur_: Utilisateur*
+ Transfert()
+ Transfert(double montant, Utilisateur* expediteur, Utilisateur* receveur_)
+ getExpediteur(): Utilisateur*
+ getReceveur(): Utilisateur*
+ getMontant(): double
+ virtual getFraisTransfert(): double
+ setMontant(double montant): void
+ setExpediteur(Utilisateur* expediteur): void
+ setReceveur(Utilisateur* receveur): void
+ effectuerTransfert(): void
+ friend operator<<(ostream& os, const Transfert& transfert): ostream&
TransfertInterac
- courriel_: string
+ TransfertInterac(double montant, Utilisateur* expediteur, Utilisateur* receveur)
+ getCourriel(): string
+ setcourriel(string courriel): void
+ virtual getFraisTransfert(): double
TransfertPaypal
- id_: string
+ TransfertPaypal(double montant, Utilisateur* expediteur, Utilisateur* receveur)
+ getId(): string
+ setId(string id): void
+ virtual getFraisTransfert(): double