UML :
DIAGRAMME D'ETATS

Le modèle dynamique représente l'évolution du système au cours du temps en réaction aux événements externes. L'évolution du système est définie par l'évolution (cycle de vie) des objets.

Le formalisme des diagrammes d'états est celui des automates. Le concept d'automate est un concept de base en informatique. Citons le livre de Hopcraft et Ullman paru en 1979 « Introduction Automata theory ». Les domaines d'application sont la programmation (spécification de programme), la compilation, et les réseaux (spécification de protocole de communication).

La notation utilisée est celle des « statechart » conçus en 1988 par David Harel. Elle a été reprise dans la méthode OMT.

Nous ne présentons dans ce cours qu'une petite partie de cette notation qui est très riche.

Les diagrammes d'états sont basés sur 3 notions :
- état d'un objet (situation d'un objet définie par ses propriétés)
- événement
- comportement des objets (leurs actions et leurs activités).

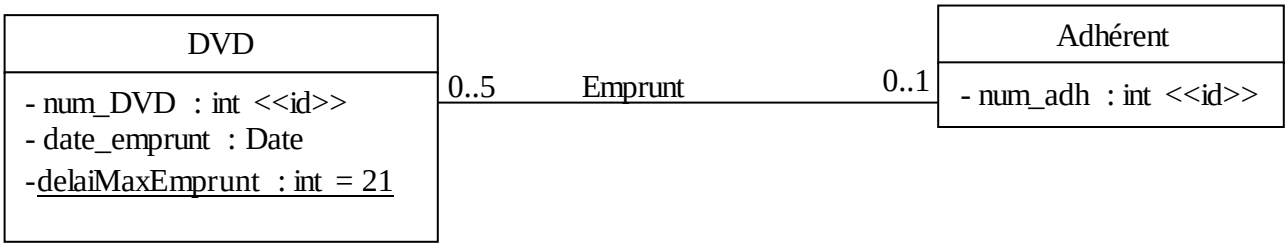
I Etat

Un état d'un objet est une situation stable dans la vie de l'objet où il effectue une activité ou il attend un événement.

Un état est caractérisé par :
- les valeurs des rubriques de l'objet
- l'existence des associations/liens de cet objet aux autres objets.

On considère uniquement les états caractéristiques d'un objet (en prenant le point de vue du système étudié).

Extrait du diagramme de classe modélisant la «gestion des emprunts de DVDs à la bibliothèque A ».



Un objet de la classe « DVD » a les états suivants : `disponible`, `emprunté`, `non rendu`. Ces 3 états sont très importants du point de vue du système étudié : le DVD est dans la bibliothèque à la disposition des adhérents, le DVD n'est plus dans la bibliothèque : il s'agit d'un emprunt en

cours ou il s'agit d'un DVD non rendu (la bibliothèque effectue des actions pour essayer de « récupérer » le DVD).

1.a Définition ou caractérisation des états

L'état « `disponible` » est défini par : date_emprunt a pour valeur NULL (non défini) et l'objet n'a pas d'association/lien de type « Emprunt ».
L'état « `emprunté` » est défini par date_emprunt < date_jour - delaiMaxEmprunt et l'objet a une association de type « Emprunt ».
L'état « `non rendu` » est défini par date_emprunt ≥ date_jour - delaiMaxEmprunt et l'objet a une association de type « Emprunt ».

Ou plus synthétiquement :

DVD disponible	DVD emprunté	DVD non rendu
date_emprunt = null	date_emprunt < date_jour - delaiMaxEmprunt	date_emprunt ≥ date_jour - delaiMaxEmprunt

Un objet est toujours dans un état connu.
A un instant donné, un objet est dans un et un seul état.

Un DVD est soit disponible, soit emprunté ou non rendu.

Les états sont stables : un objet est dans un état donné pour un certain temps (une durée non négligeable) relativement à l'échelle de temps du système étudié.

I.b Etats de la classe Facture

Facture
- num_facture : int <<id>> - date_paiement : Date = NULL - date_relance: Date = NULL -nombreFactures: int = 0 - montant_Facture :float

Extrait du diagramme de Classes

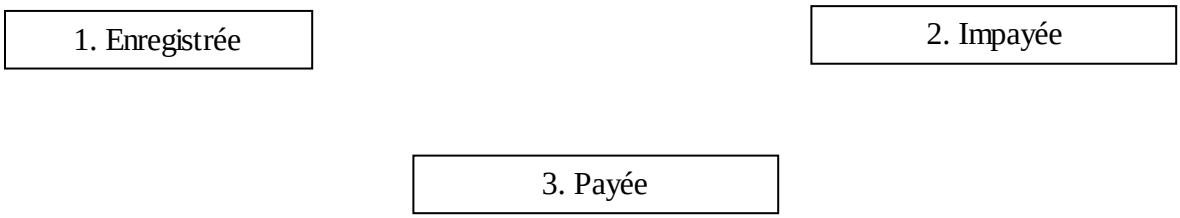
Un objet de la classe « Facture » a les états suivants : `enregistrée, impayée` et `payée`. C'est trois états sont très importants de point de vue du système étudié (entreprise C) : la facture a été envoyée et on attend le paiement, la facture n'a pas été payée dans le délai et la facture a été payée.

Définition ou caractérisation des états
L'état « `enregistrée` » est défini par date_relance = NULL et date_paiement = NULL.
L'état « `Impayée` » est défini par date_relance ≠ NULL et date_paiement = NULL.
L'état « `Payée` » est défini par date_paiement ≠ NULL.

caractérisation des états synthétique :

Facture enregistrée	Facture payée	Facture impayée
date-paiement =NULL date_relance= NULL	date_paiement ≠ NULL	date-paiement =NULL date_relance ≠ NULL

Représentation graphique des états de la classe « Facture » :



Dans certain cas, les états des objets d'une classe ne peuvent pas être caractérisés par les valeurs des rubriques de l'objet ou l'existence des associations de cet objet aux autres objets. Dans ce cas-là, la classe a une rubrique supplémentaire « etat_classe » dont la valeur sera l'état de l'objet. Un four a deux états : en_marche et inactif. Ces deux états sont enregistrés au travers de la rubrique etat_four. Les valeurs des autres rubriques ne peuvent pas aider à déterminer l'état du four. On a donc ajouté une rubrique supplémentaire : etat_four.

Four
- etat_four : Boolean - mode_cuisson : String - temps_minuteur : Integer = 0

Si un objet passe par plusieurs états, on dit que l'objet a un cycle de vie. Tous les objets d'une même classe ont des cycles de vie qui ont la même structure. Cette structure est définie par le diagramme d'états associé à la classe.

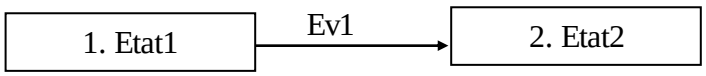
Il y a des classes sans diagramme de classe : leurs instances n'ont pas de cycle de vie. Seule une minorité de classes ont des cycles de vie.
Un diagramme d'états est toujours associé à une classe.

Un changement d'état a pour origine un événement (appel d'une méthode de la classe).
L'exécution d'une méthode ne change pas toujours l'état de l'objet.

II Transition/Evénement

Une transition est le passage d'un état à un autre état. Une transition est représentée par une flèche de l'état d'origine au nouvel état. Une transition est étiquetée par l'événement qui provoque le changement d'état.

Toute transition est étiquetée par un événement (nom d'une méthode)



II.a Evénement

Il y a 5 types d'événements :

Les événements externes : événement produit par un acteur et destiné à un objet du système (instance d'une classe).

- Un événement externe correspond à une opération/méthode de la classe de l'objet destinataire.
- Le nom d'un tel événement est préfixé par ARR_. Exemple : ARR_paiement, ARR_dde_DVD, ARR_commande.
- Un événement externe peut être l'étiquette d'une transition dans le diagramme d'états associée à la classe des objets qui reçoivent cet événement.

Les événements résultats : événement produit par un objet du système à destination d'un acteur externe.

- Le nom d'un tel événement est préfixé par « ENV ». Exemple : ENV_facture, ENV_relance
- Un événement résultat ne peut pas être étiquette d'une transition dans un diagramme d'états.

Les événements temporels : événement signalisant l'arrivée d'une échéance temporelle ; ils sont générés à la fin d'un délai d'attente ou à l'arrivée d'une date précise (date qui déclenche l'exécution de tâche par le système). Ces événements sont non porteur d'information : ils n'ont pas d'argument.

- Un événement temporel correspond à une opération/méthode de la classe de l'objet destinataire.
- Le nom d'un événement temporel est par exemple : ARR_fin_délai_de_paiement(), ARR_fin_délai_livraison(), ARR_date_facturation().
- Un événement temporel peut être l'étiquette d'une transition dans le diagramme d'états associée à la classe des objets qui reçoivent cet événement.

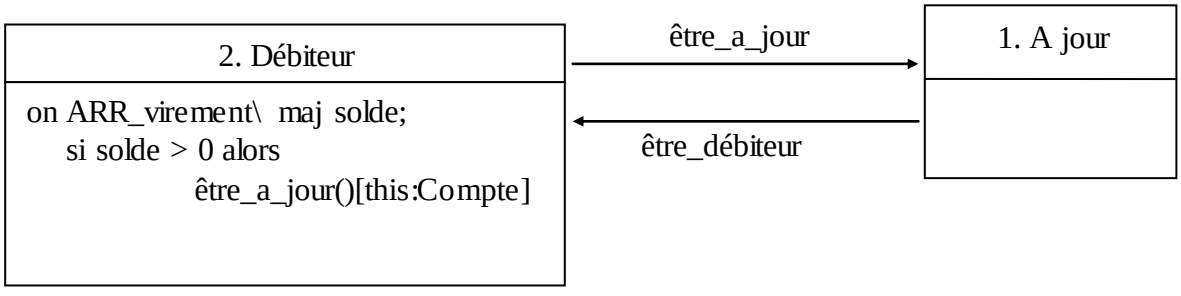
Les événements modificateurs : événement produit par un objet du système à destination d'un autre objet du système.

- Un événement modificateur correspond à une opération/méthode de la classe de l'objet destinataire.
- Il n'y a pas de convention sur le nom d'un événement modificateur.
- Un événement modificateur peut être l'étiquette d'une transition dans le diagramme d'états associée à la classe des objets qui reçoivent cet événement.

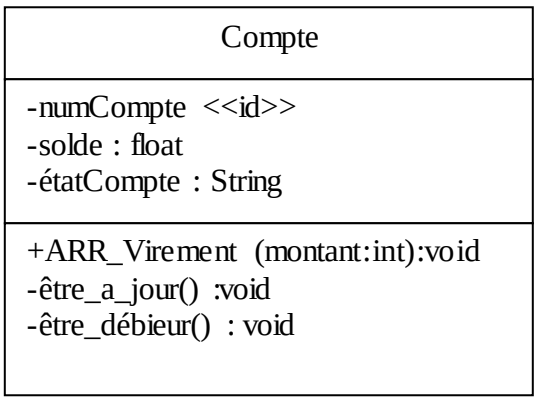
Les événements propres : événement généré par un objet pour lui-même

- Un événement propre peut être l'étiquette d'une transition dans le diagramme d'états associée à la classe des objets qui reçoivent cet événement.
- Un événement propre correspond à une opération/méthode privée de la classe de l'objet.

Extrait du diagramme d'états de la classe Compte (événement « être_a_jour » est un événement propre).

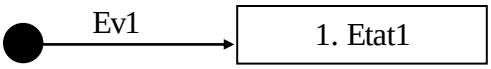


Extrait du diagramme de classes

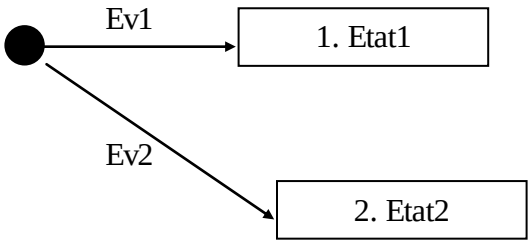


II.b Evénements particuliers

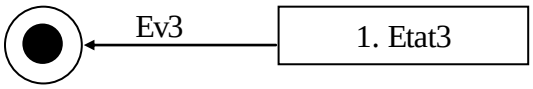
Dans le diagramme suivant, E1 est un événement créateur. Il crée d'une instance de la classe, instance qui est dans l'état1 :



Une classe peut avoir plusieurs événements créateurs et plusieurs états initiaux (premiers états d'un objet).

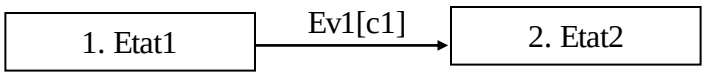


Dans le diagramme suivant, E3 est un événement destructeur. Son occurrence détruit l'objet (seulement, si il était dans Etat3).



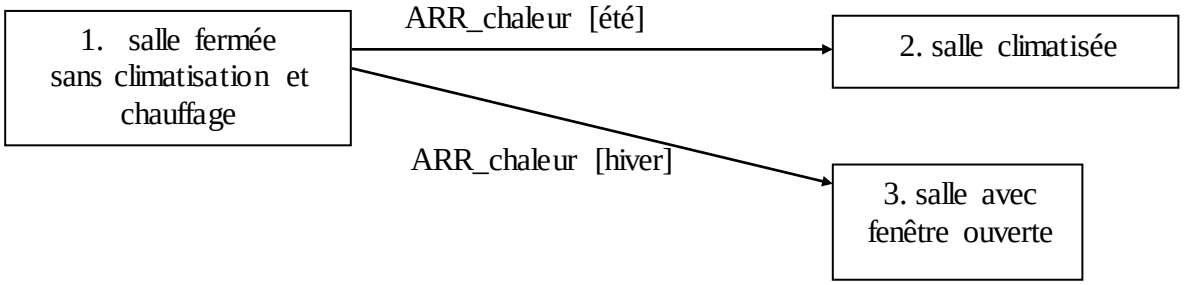
Une classe peut avoir plusieurs événements « destructeur ».

II.c Garde sur les transitions



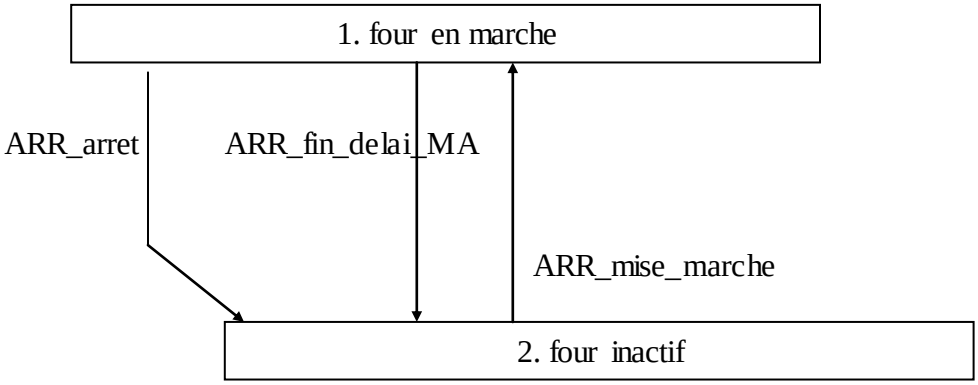
D'après l'extrait du diagramme d'états précédant : de l'état1, un objet passe à l'état2 si et seulement si

- L'événement Ev1 a eu lieu et
- La condition c1 est satisfaite.



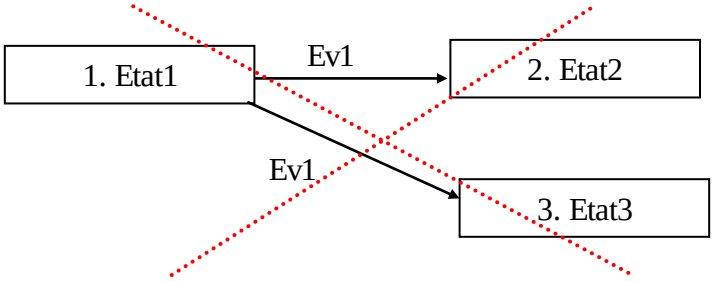
II.d Propriétés des événements

- Autonomie des objets : un objet récepteur d'un événement réagit ou non à l'événement en fonction de son état. Dans le cas du diagramme d'états suivant, si le four est dans l'état en marche, il ignore l'événement ARR_mise_marche ; événement pris en compte lorsqu'il est dans l'état inactif.
  Il est donc inutile d'arrêter l'horloge lorsqu'un objet de la classe « Four » passe dans l'état « four inactif » car l'événement produit par l'horloge sera ignoré.



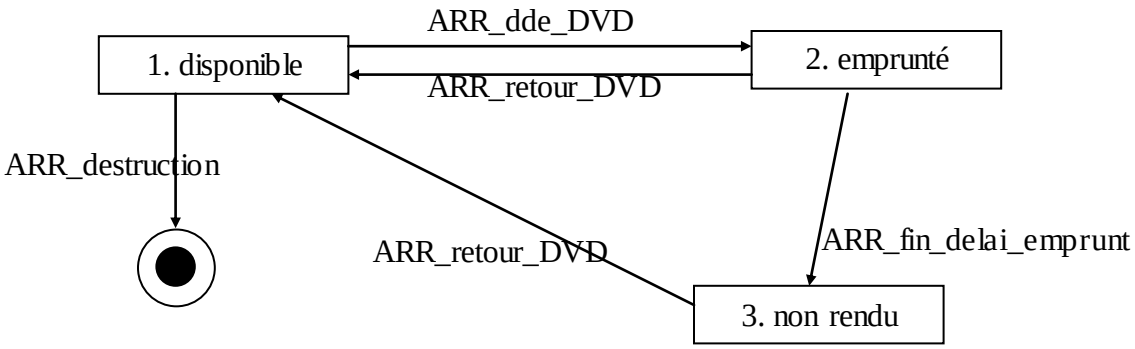
- Asynchronisme des objets : L'objet émetteur d'un événement continue ses traitements indépendamment des traitements réalisés par l'objet récepteur de l'événement. L'objet émetteur d'un événement ne préjuge pas des traitements que l'objet récepteur va réaliser et n'attend aucun retour de l'objet récepteur. L'émission d'un événement peut être comparé à l'envoi d'une lettre : on est sûre que le destinataire recevra, lira et traitera la lettre ; mais on ne sait pas quand ; et durant ce temps l'émetteur continue ses activités.
  Un message synchrone peut être comparé avec un coup de fils. La personne qui appelle attend au bout du fils que le destinataire réponde et que la communication soit terminée avant de reprendre ses autres activités.

Un diagramme d'états est déterministe : il n'y a pas en sorti d'un même état deux transitions ayant la même étiquette (déclenchées par le même événement).

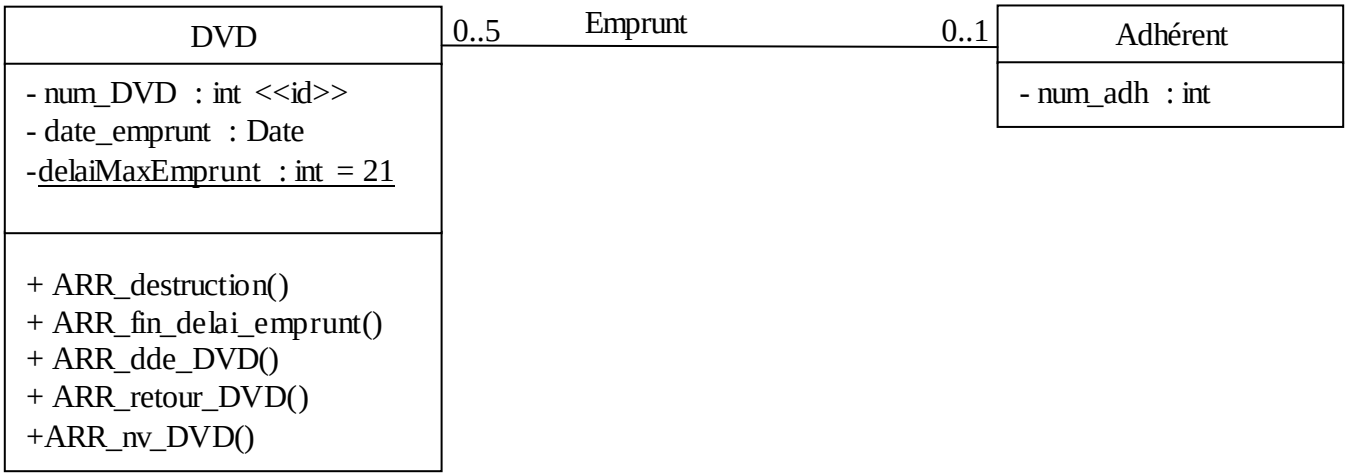


II.e Cohérence avec le diagramme de classe

Extrait du diagramme d'états de la classe « DVD » dans le cadre de l'analyse des «emprunts à la bibliothèque A ». Ce diagramme comprend plusieurs transitions avec le même événement déclencheur (ARR_retour_DVD).



Un événement lié à une transition du diagramme d'états de la classe A est associé à une opération/méthode de la classe A (apparaissant dans la définition de la classe A dans le diagramme de classes).
Donc le diagramme de classes, une fois complété d'après l'extrait du diagramme d'états de la classe « DVD » est :



III Activités/Traitements

Les diagrammes d'états seraient de peu d'utilité s'ils se bornaient à décrire la structure du cycle de vie des objets sans détailler l'aspect comportemental des objets : ce que font les objets en réponse aux événements.

Le comportement d'un objet prend deux formes :
L'activité – opération associée à un état qui nécessite un certain temps et qui peut être interrompue (terme clef *do*) Les activités comprennent :

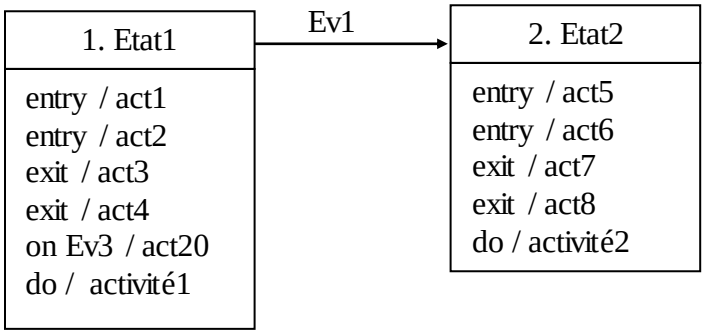
- des opérations continues comme l'affichage des images prises par une caméra de surveillance ou comme le contrôle d'une sonnerie de téléphone.
- des opérations séquentielles qui se terminent par elles-mêmes comme la gestion d'un déplacement de robot (activité qui se terminera lorsque le robot aura fini son déplacement) ou comme l'exécution de calcul important.

L'arrivée d'un événement qui doit être prise en compte par l'objet interrompt son activité en cours.

Les traitements – ensemble d'actions qui ne peuvent pas être interrompues. A chaque état, on peut associer 3 types de traitement.

- Traitement en entrée : ensemble d'actions exécutées par tout objet qui rentre dans cet état (terme clef : *entry*)
- Traitement en sortie : ensemble d'actions exécutées par tout objet qui sort de cet état (terme clef : *exit*)
- Traitement associé à un événement : ensemble d'actions exécutées par tout objet qui reçoit cet événement (terme clef : *on « nom de l'événement »*).

III.a Illustration du mécanisme d'exécution des actions et des activités

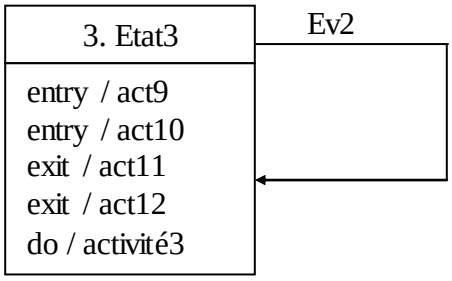


Extrait du diagramme d'états de la classe A

D'après l'extrait du diagramme d'états de la classe A précédent, un objet de la classe A dans l'état « Etat1 » lorsqu'il reçoit l'événement Ev1 :

- arrête l'activité activité1,
- exécute les actions act3 et act4, (le traitement associé à la sortie de l'état Etat1)
- change d'état : il quitte l'Etat1 et va dans l'Etat2.
- Exécute act5 et act6 (le traitement associé à l'entrée dans l'état Etat2),
- commence l'activité activité2.

Un objet de la classe A dans l'état « Etat1 » exécute l'action act20 lorsque la méthode « Ev3 » est exécutée.



Extrait du diagramme d'états de la classe B

D'après l'extrait du diagramme d'états de la classe B précédant, un objet de la classe B dans l'état « Etat2 » lorsqu'il reçoit l'événement Ev2 exécute les actions suivantes :

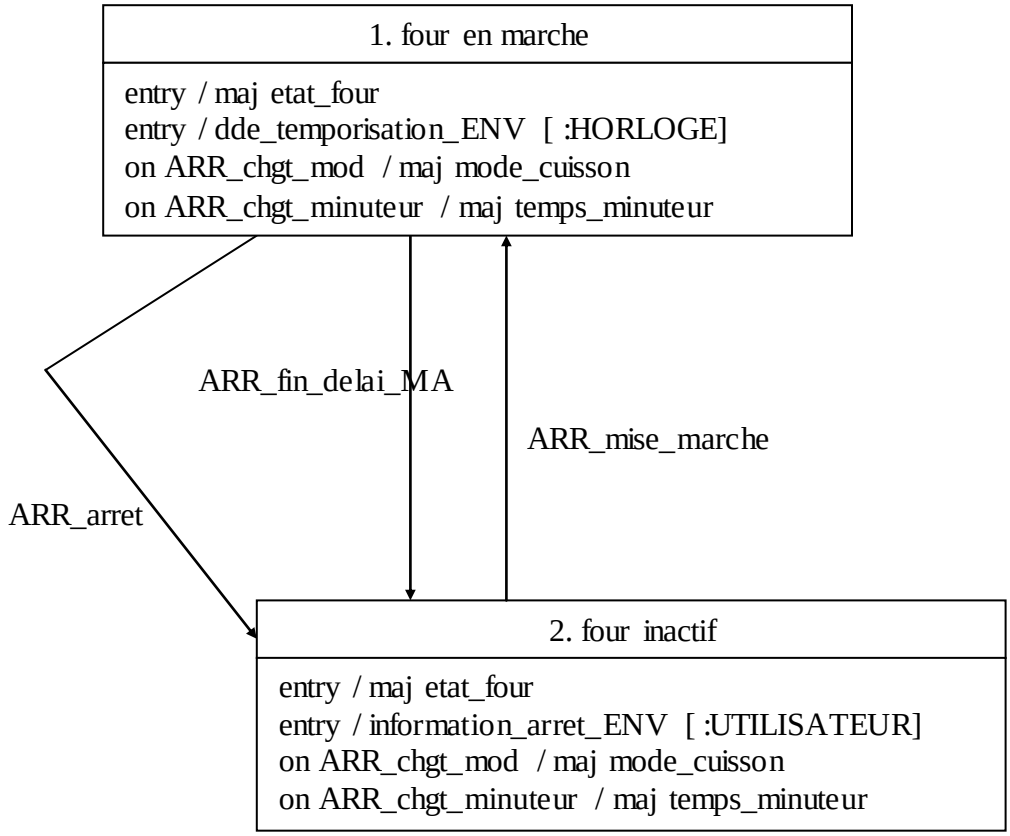
arrêt de l'activité3, act11, act12, act9 et act10, commencer l'activité3

Il s'agit du traitement associé à la sortie et le traitement associé à l'entrée de l'état Etat3.

Les traitements ne peuvent pas être interrompus : un objet finit un traitement avant de traiter les événements qui sont en attente.

Un objet effectue un seul traitement/activité à la fois. Plusieurs objets (de la même classe ou non) peuvent effectuer des traitements et/ou activités en parallèle. Exemple : deux objets de la classe A peuvent traiter chacun indépendamment un événement de type Ev1 (il ne s'agit pas du même événement, mais de deux événements de même type) pendant qu'un objet de la classe B traite un événement Ev2.

Le diagramme suivant est le diagramme d'états de la classe « Four » dans le cadre de la modélisation du fonctionnement d'un four à micro-onde. Notons que les événements ARR_chgt_mode, ARR_chgt_minuteur ne provoquent pas de changement d'état; mais, nécessitent un traitement particulier qui est possible quel que soit l'état du four ; c'est pourquoi, les actions à exécuter sont précédées de la clef *on « nom_eve »*.



Un traitement est une série d'actions.

III.b actions

Il y a 6 types d'actions :

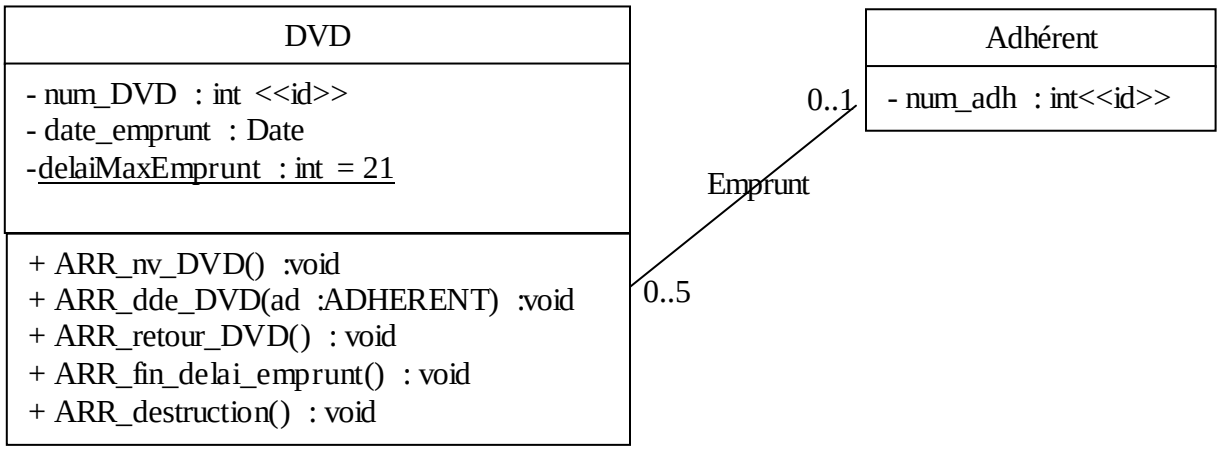
- Création d'une instance d'une classe (d'un objet) : *créer inst. « nom de la classe »*.
  Exemple : créer inst. DVD.
  Vérifiez que la classe est bien une classe du diagramme de classes.
- Création d'une instance d'une association. Syntaxe : *créer inst. de l'ass. « nom de l'association »*
  Exemple : créer inst. de l'ass. Emprunt.
  Vérifiez que l'association est dans le diagramme de classes.
- Détruire une instance d'une association. Syntaxe : *détruire inst. l'ass. « nom de l'association »*
  Exemple : détruire inst. l'ass. Emprunt.
  Vérifiez que l'association est dans le diagramme de classes.
- Détruire une instance d'une classe (objet). Syntaxe : *détruire « nom de la classe »*
  Exemple : créer détruire DVD.

Vérifiez que la classe est bien une classe du diagramme de classes.

- Mise à jour de la valeur des rubriques d'une classe ou d'une association. Syntaxe : *maj « nom de la rubrique »*
  Exemple : maj date_emp = NULL ; maj mode_cuisson.
  Vérifiez qu'il s'agit bien des rubriques d'une classe ou d'une association (attention au nom des rubriques).
- Effectuer des calculs et diverses opérations (dont le calcul de valeur de rubriques de nature « code »)
  Exemple : calcul et maj num_DVD ; calculer le total de la facture ; choisir la voiture louée.
- Générer des événements. Syntaxe *: «nom de l'événement »[type du destinataire]* (le type du destinataire est dans le format : *nom_objet :nom_classe* ou *nom_objet :nom_acteur* .
  Exemple : ENV_relance [:ADHERENT], ENV_ information_pret [:ADHERENT], a_detruire [lui_même :Client], virement[:Compte Bancaire]
  Vérifiez que le destinataire peut recevoir l'événement : si le destinataire est un objet, l'événement doit être une opération/méthode de sa classe.

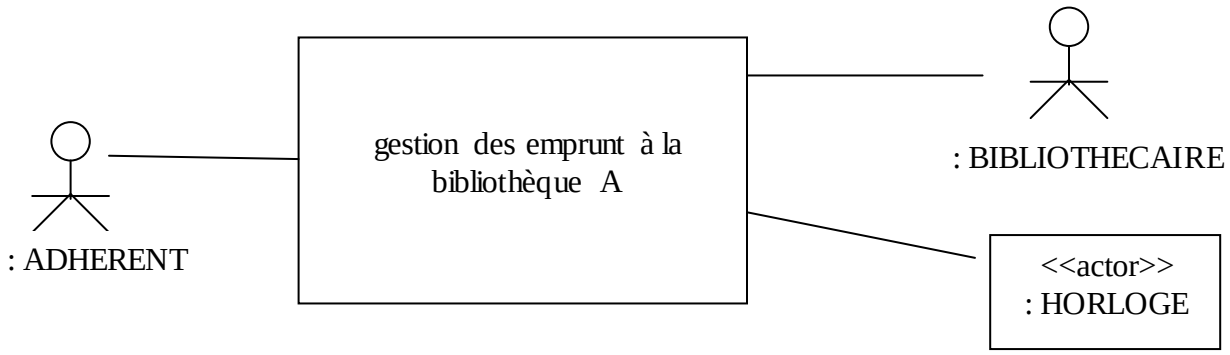
  Les événements à destination des acteurs externes peuvent demander une préparation (consultation de la base de données). Si on ne veut pas détailler l'ensemble des opérations nécessaires à l'envoi de l'événement (édition d'un document) alors on regroupe l'ensemble de ces opérations sous le terme : préparer l'envoi de l'événement xxx ou préparer l'événement xxx à envoyer.

III.b.1 Exemple « bibliothéque A »

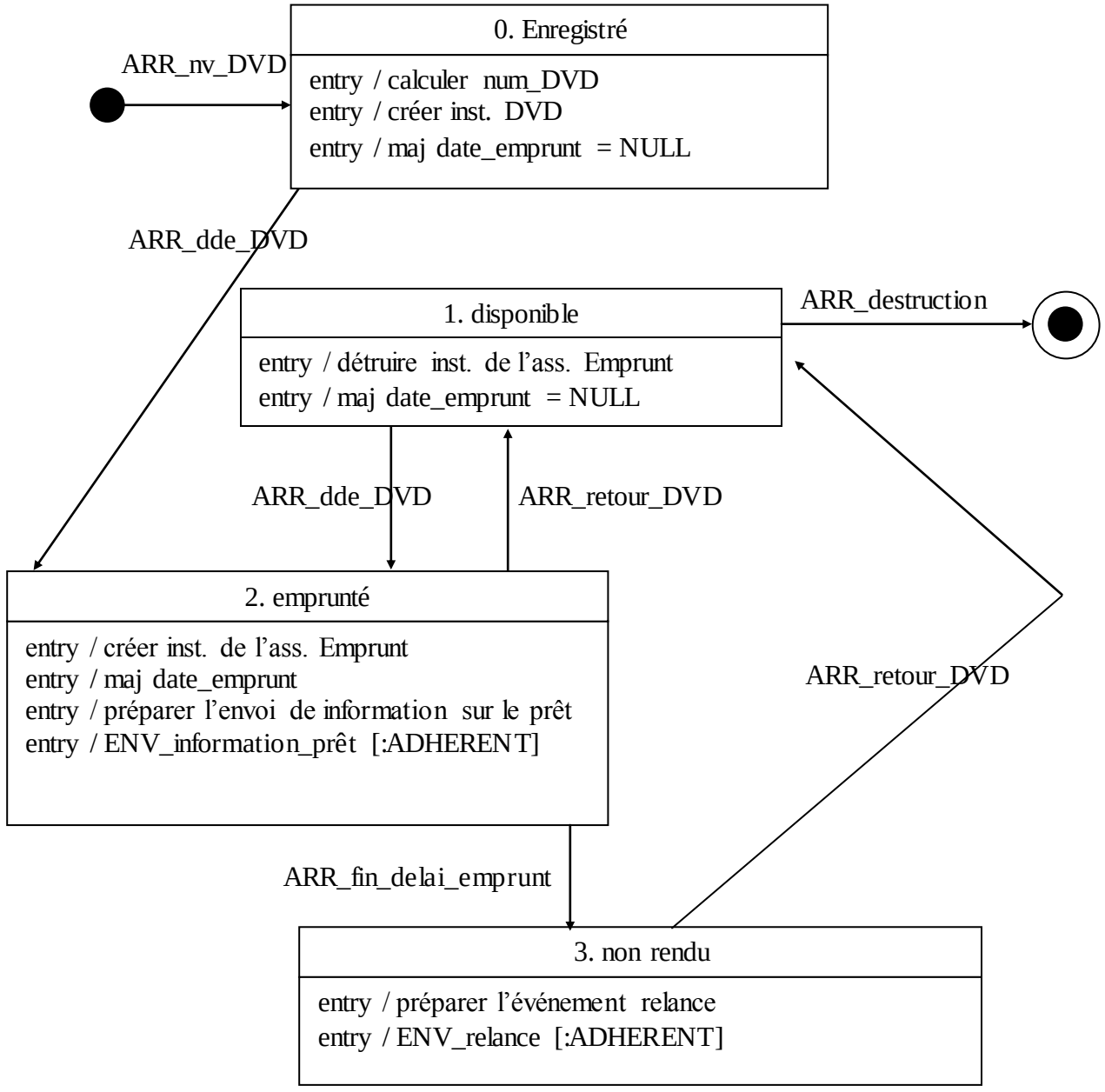


Notons que l'opération/méthode ARR_dde_DVD a besoin d'un paramètre, un objet de la classe « Adhérent », pour créer l'association Emprunt avec l'adhérent emprunteur.

En accord avec le diagramme d'états de la classe « DVD», le diagramme de contexte statique de la «gestion des emprunts à la bibliothèque A » est :



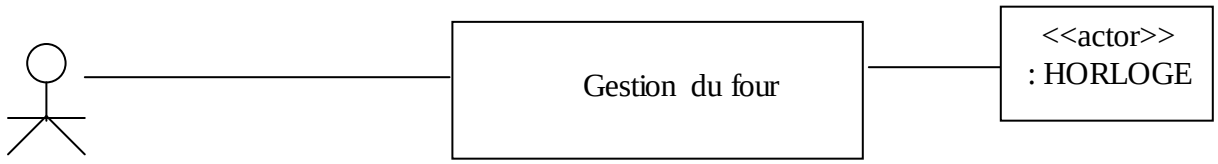
Voici le diagramme d'états de la classe « DVD » dans le cadre de l'analyse de la «gestion des emprunts à la bibliothèque A ».



Notons que nous avons ajouté un état (0. Enregistré). En entrée de cet état, les actions nécessaires à la création d'une instance de DVD sont exécutées. Ces actions sont exécutées une fois dans la « vie » d'un objet. Alors que les actions exécutées en entrée de l'état « disponible » sont exécutées à chaque retour du DVD à la bibliothèque.

III.b.2 Exemple « four à micro_onde »

En accord avec le diagramme d'états de la classe « Four », le diagramme de contexte statique du fonctionnement d'un four à micro-onde est :



En accord avec le diagramme d'états de la classe « Four », le diagramme de classe modélisant du fonctionnement d'un four à micro-onde est :