
GUIDE DE L'ÉTUDIANT POUR LES DIAGRAMMES D'ACTIVITÉS

Domingo Palao Muñoz
Charles-Antoine Brunet

23 septembre 2020

Version : A20-01 (2020-09-23)

Ce document est réalisé avec l'aide de L^AT_EX et de la classe gegi-minidoc.

©2020 Tous droits réservés. Département de génie électrique et de génie informatique,
Université de Sherbrooke.

Liste des figures

1	Exemple avec <i>pins</i>	3
2	Exemple : réveil le samedi	3
3	Exemple : compter et afficher	4
4	Exemple : compter le nombre de valeurs positives	4
5	Exemple : test automatisé	5

Liste des tableaux

1	Éléments d'un diagramme d'activités	2
---	---	---

1 Introduction

UML est une notation graphique conçue pour représenter, spécifier, construire et documenter les systèmes logiciels. Ses deux principaux objectifs sont la modélisation de systèmes utilisant les techniques orientées objet depuis la conception jusqu'à la maintenance, et la création d'un langage abstrait compréhensible par l'être humain et interprétable par les machines.

UML s'adresse à toutes les personnes chargées de la production, du déploiement et du suivi de logiciels (analystes, développeurs, chefs de projets, architectes, etc.), mais peut également servir à la communication avec les clients et les utilisateurs du logiciel. Il s'adapte à tous les domaines d'application et à tous les supports. Il permet de construire plusieurs modèles d'un système, chacun mettant en valeur des aspects différents : fonctionnels, statiques, dynamiques et organisationnels.

UML propose 13 diagrammes et est devenu un langage incontournable dans les projets de développement. Il faut savoir que UML n'est pas une méthodologie, c'est un langage qui permet la communication. Pour cette raison, l'utilisation d'un ou plusieurs diagrammes est votre responsabilité. Dans ce document, seulement le diagramme d'activités est abordé. Les éléments présentés dans ce guide sont inspirés et tirés de deux références principales [1, 2].

2 Diagramme d'activités

Les diagrammes d'activités sont classés dans le groupe de diagrammes comportementaux, c'est-à-dire, qu'ils aident à modéliser et comprendre le comportement d'une situation donnée. Ils sont une combinaison de différentes techniques comme les diagrammes d'événements, les techniques de modélisation d'états, les réseaux de Petri, etc. Ces diagrammes sont particulièrement utiles.

La vision des diagrammes d'activités est centrée sur les flots de contrôle. On y trouve trois éléments fondamentaux :

- Les activités qui sont représentées par un rectangle aux coins arrondis. Elles décrivent un traitement. Le flot de contrôle reste dans l'activité jusqu'à ce que les traitements soient terminés. On peut définir des variables locales à une activité et manipuler les variables accessibles depuis le contexte de l'activité. Il n'est pas nécessaire de déclarer les variables, même s'il est possible de les manipuler.
- Les transitions qui sont représentées par des flèches aux traits pleins qui connectent les activités entre elles. Les transitions sont déclenchées dès que l'activité source est terminée et déterminent la prochaine activité à déclencher. Contrairement aux

activités, les transitions sont franchies de manière atomique, en principe sans durée perceptible.




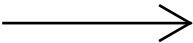
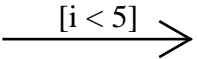
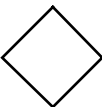


- Les décisions qui sont représentées par des losanges. Une décision reçoit une condition (garde) qui sera évaluée et produira un résultat booléen (vrai ou faux). Le flux de contrôle empruntera le chemin correspondant au résultat produit par l'évaluation de la condition.

Chaque activité peut être suivie d'une autre activité; c'est alors un séquençement simple. Le diagramme d'activités permet donc d'indiquer l'ordre des choses. Un diagramme d'activités permet aussi de montrer des activités en parallèle, c'est cela la plus grande différence par rapport aux ordinogrammes (*flow charts*).

3 Éléments d'un diagramme d'activités

Le tableau suivant montre les éléments qui peuvent composer un diagramme d'activités avec une brève description.

TABLEAU 1 : Éléments d'un diagramme d'activités

Élément	Description	Représentation
Point de départ	Indique le point de départ de toutes les activités, là où débute l'algorithme.	
Point de finalisation	C'est le point de fin. C'est là que la séquence d'activités se termine, la fin de l'algorithme.	
Activité	Une boîte avec les coins arrondis représente l'exécution d'une activité.	
Transition	Une flèche représente une transition qui est déclenchée par la fin d'une activité et qui provoque le début immédiat de la suivante.	
Transition avec garde	Une transition peut être accompagnée d'une garde (une condition) entre crochets. La transition ne peut être alors empruntée que si la garde est évaluée à vrai.	
Point de jonction	Un losange peut être utilisé pour mettre en valeur le branchement conditionnel. Il peut y avoir un nombre quelconque de transitions en entrée et en sortie. Les transitions de sorties sont mutuellement exclusives et une seule peut être vrai à tout moment.	
Pin	Une <i>pin</i> spécifie une entrée ou une sortie.	
Note	Une courte note textuelle explicative peut être ajoutée à un diagramme.	

Il est intéressant de noter qu'il n'y a aucun symbole pour illustrer une itération, comme un *tant que* (*while*) en pseudocode. Une itération est réalisée en créant un cycle (boucle) dans le diagramme, comme le montre les figures 3 et 4. Pareillement, il n'y a pas de symbole pour illustrer une fonction, la section suivante discute de ce sujet.

4 Flot des données et fonctions

Le déroulement se montre bien avec un diagramme d'activités, mais le flot des données est parfois moins bien explicité. Le flot des données peut être montré dans un diagramme d'activités avec l'aide de *pins*. Les *pins* permettent de définir les entrées et les sorties et donc une interface. La figure suivante montre l'interface avec *Calculer somme*.

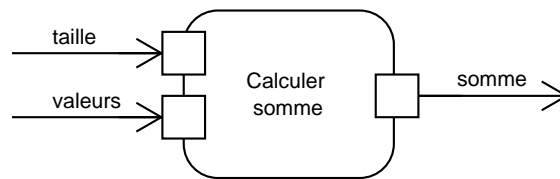


FIGURE 1 : Exemple avec *pins*

Les *pins* peuvent être interprétées comme étant une interface avec une fonction, comme le montre l'exemple à la section 5.3. Les *pins* en entrée peuvent être des paramètres et les pins en sortie peuvent être la valeur de retour ou des paramètres en sortie. Cette interprétation des *pins* a ses limites et il ne faut pas pousser l'analogie avec une fonction trop loin.

5 Exemples

5.1 Réveil le samedi

La figure 2 montre un exemple qui représente la séquence d'activités le premier samedi après la fin de la session.

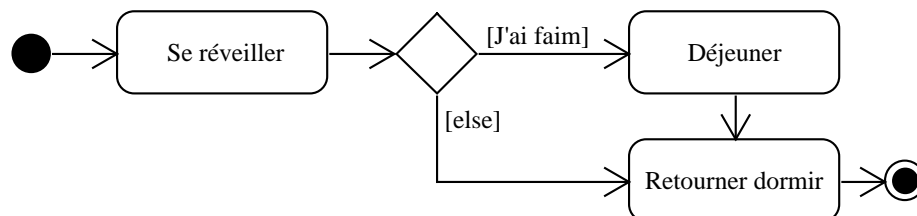


FIGURE 2 : Exemple : réveil le samedi

5.2 Compter et afficher

La figure 3 montre un exemple avec une boucle et qui termine en affichant un message. Il est intéressant aussi de noter qu'il n'y a pas d'activité pour la déclaration de variables. S'il est pertinent de donner des détails sur les variables, une note peut être ajoutée.

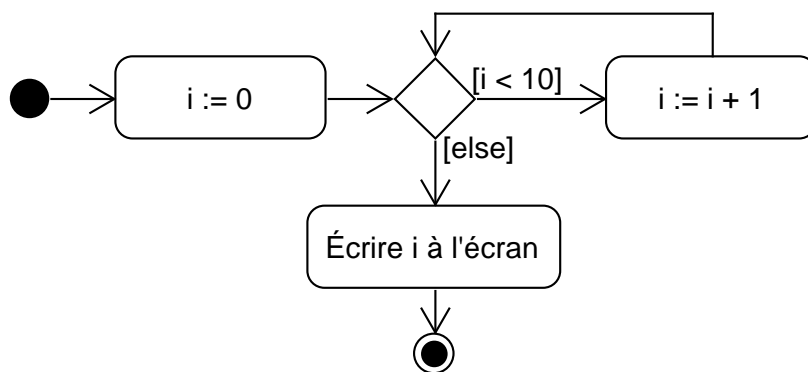


FIGURE 3 : Exemple : compter et afficher

5.3 Nombre de valeurs positives

La figure 4 montre un exemple d'algorithme qui calcule le nombre de valeurs positives dans un tableau. Il est à noter que la première activité contient plusieurs initialisations de variables. Cela est permis tant que les opérations sont très similaires et que la clarté du diagramme n'en souffre pas.

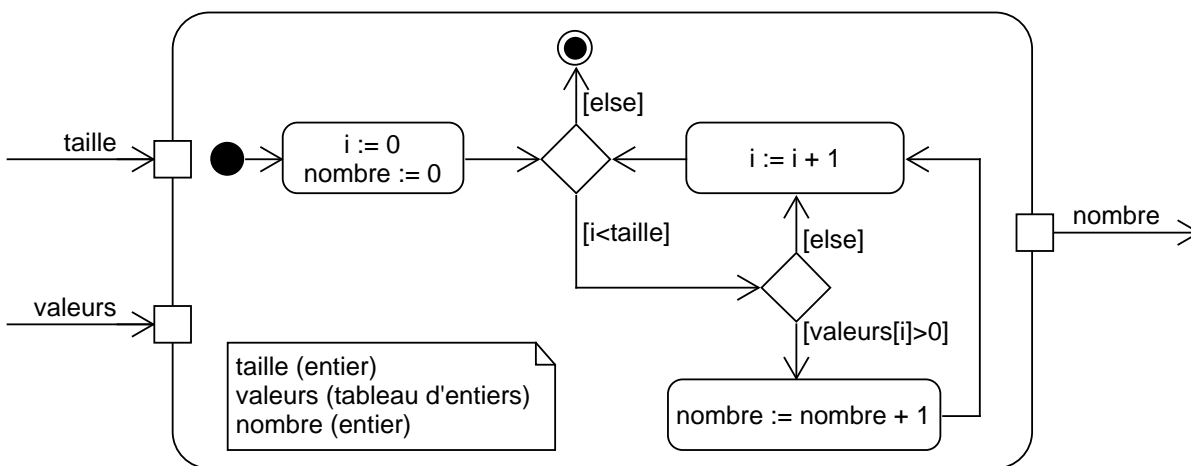


FIGURE 4 : Exemple : compter le nombre de valeurs positives

5.4 Un test automatisé

La figure 5 montre un exemple d'un algorithme qui fait un test automatisé de l'activité de la figure 4.

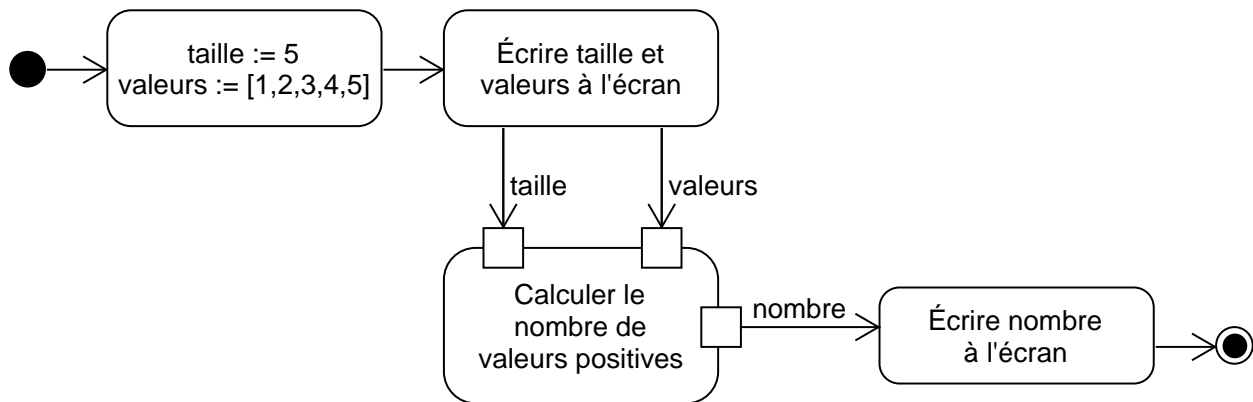


FIGURE 5 : Exemple : test automatisé

Liste des références

- [1] B. Charroux, A. Osmani, et Y. Thierry-Mieg, *Langage UML 2 : Pratique de la modélisation*, 3^e édition, séries Collection Synthex. Pearson Education, 2010.
- [2] M. Fowler, *UML Distilled : A Brief Guide to the Standard Object Modeling Language*, 3^e édition. Addison-Wesley, 2003.