C2.6 – Activités, Tâches et Sous-processus

Le tout premier concept que nous avons présenté dans ce cours est les activités. Une activité est une action, une unité de travail réalisée aux cours d'un processus, avec un début et une fin bien identifiée. Nous allons dans cette partie du cours détailler cette notion d'activité en présentant les différents types d'activités que l'on peut rencontrer dans la norme BPMN.

1. TACHE

Commençons par la notion de tâche. Une tâche est un type d'activité qui ne peut être décomposée, c'est-à-dire qu'il n'est pas possible de définir un niveau de détail plus fin. On parle d'unité élémentaire ou atomique.
Par définition, une tâche étant dédiée à une action bien particulière, il est possible d'en spécifier sa nature. Comme pour les autres concepts de la norme, nous pouvons ajouter des pictogrammes en haut à gauche de l'activité pour définir le type de tâche.

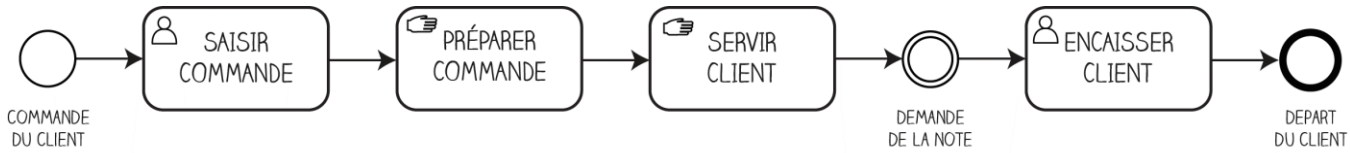
On distingue notamment la tâche de ***réception*** qui spécifie que l'on reçoit un message d'un utilisateur extérieur et la tâche ***d'envoi*** qui spécifie que l'on envoie un message à un utilisateur externe au processus. On remarque ici que les icônes suivent la même règle que pour les évènements de type « catch » et « throw » c'est-à-dire avec l'enveloppe pleine pour une tâche d'émission et vide pour une tâche de réception.

La tâche ***utilisateur*** précise que la tâche est réalisée par un acteur humain mais interagissant avec une application informatique, contrairement à la tâche ***manuelle*** qui est réalisée exclusivement par un acteur humain.

La tâche de ***service*** est une tâche automatisée, c'est-à-dire sans intervention humaine. L'application informatique déclenchée est vue comme un service demandé. On ne connait pas le contenu de l'application, on ne connait que les résultats produits. La tâche de ***script*** est également une tâche automatisée mais dont le comportement a été construit spécifiquement pour la gestion du processus. C'est donc un type de tâche qui ne concerne que ceux qui souhaiteraient à terme créer un moteur pour la gestion automatique de leur processus.

Enfin, la tâche de ***règle de gestion*** permet d'indiquer que l'action de la tâche est d'appliquer une règle métier pour prendre une décision.

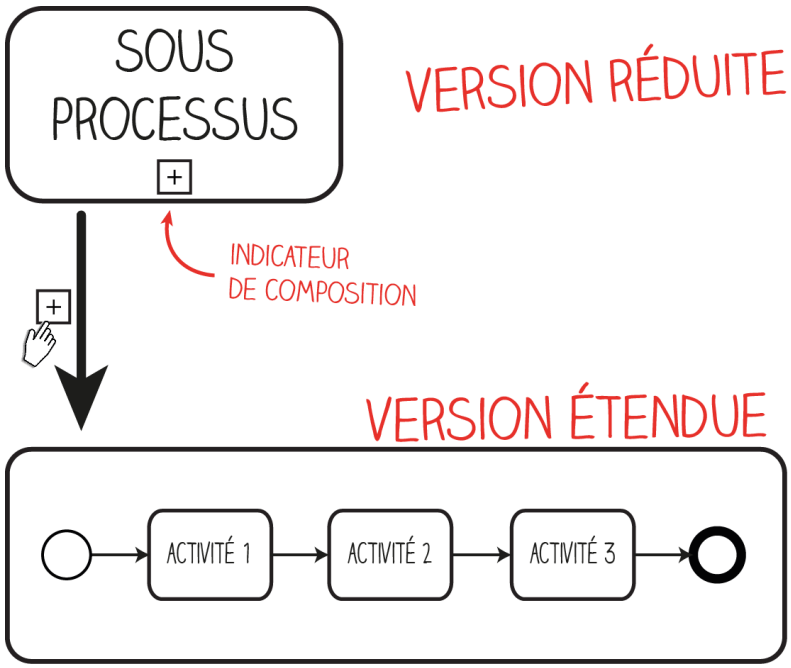
Dans notre processus simple de service au client dans un restaurant, la saisie de la commande serait une tâche utilisateur, tout comme la tâche de l'encaissement. Les tâches de préparation de la commande et de service du client seraient des tâches exclusivement manuelles.



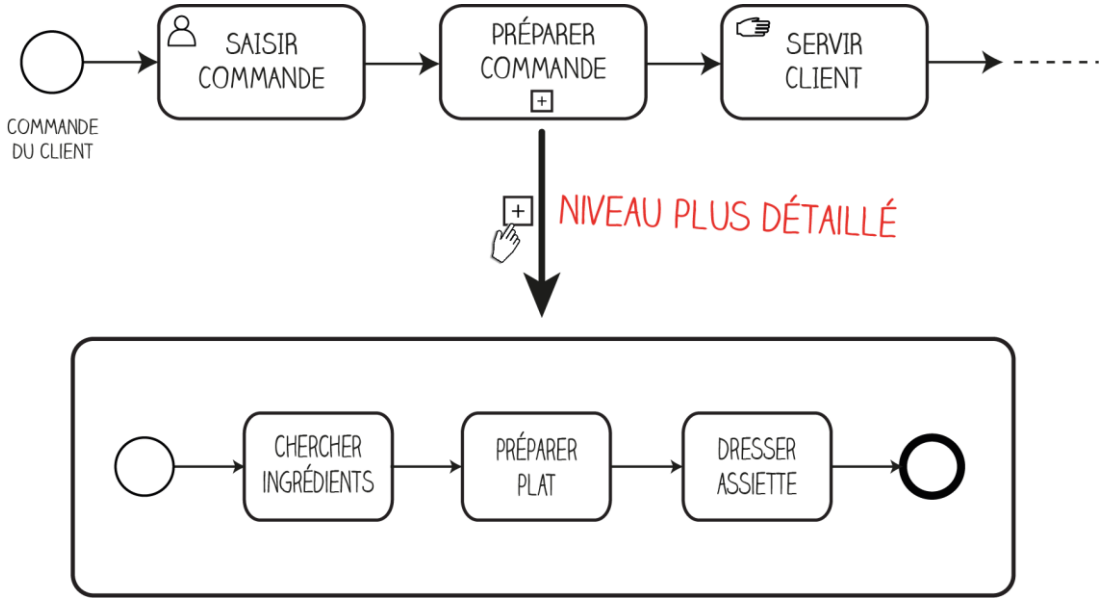
2. LES SOUS-PROCESSUS

À l'inverse de la tâche, un sous-processus est une activité composée, c'est-à-dire une activité qui peut elle-même être décrite suivant une séquence d'activités.

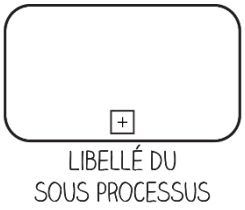
Le sous-processus peut être représenté de deux façons : soit dans une **version réduite** ; un marqueur carré avec le signe + est alors inscrit sur le sous-processus. Ce signe ne doit pas être confondu avec la notion de parallélisme que l'on a pu voir avec les évènements ou les passerelles. Il permet simplement de spécifier le fait que cette activité est composée.
Dans le diagramme, le sous-processus peut également être affiché dans sa **version étendue**. La forme de l'activité est alors élargie afin de laisser la place d'inscrire la séquence du sous-processus à l'intérieur.



Dans notre exemple, on pourrait facilement dire que l'activité « préparer commande » est un sous-processus, c'est-à-dire qu'elle contient elle-même une séquence d'activités.
Si l'on étend l'affichage du sous-processus, nous pourrions obtenir un flux d'activités composé d'un évènement de départ, de différentes tâches « chercher ingrédients », « préparer plat » et « dresser assiette », flux qui se termine par un évènement de fin.



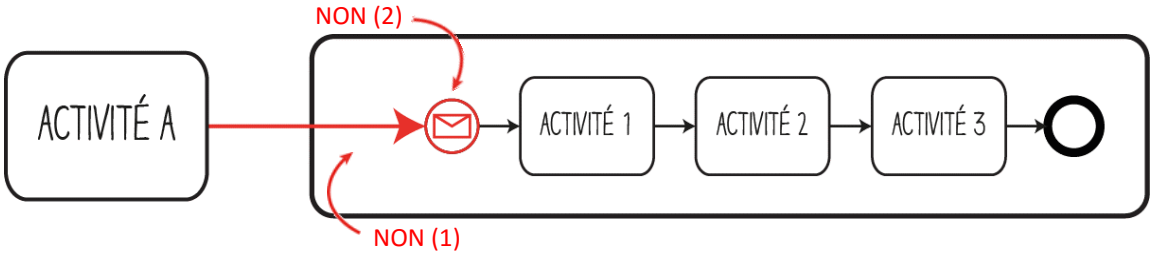
Un sous-processus est donc un processus à part entière. En termes de notation, son libellé est noté en dessous du rectangle. De plus, contrairement à celui d'une tâche exprimé par un verbe, le libellé d'un sous processus est un nom, généralement le substantif d'un verbe.



Par exemple, l'activité « préparer commande » devient le sous-processus « préparation de la commande ».

Il y a deux autres règles à respecter concernant les sous processus :

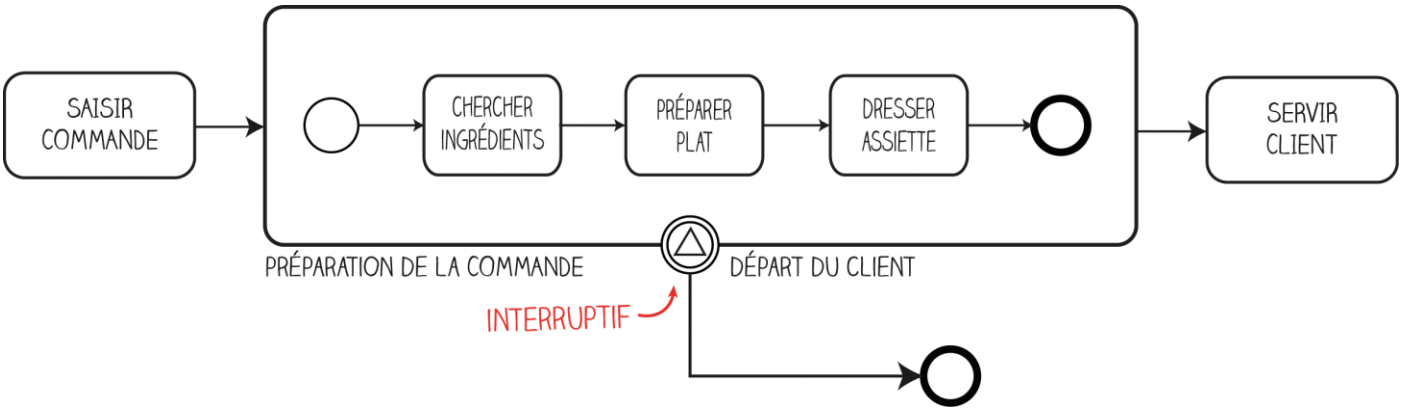
1. Le flux de séquence ne peut pas passer au-delà de la frontière du sous-processus. Le flux entrant et sortant doit être connecté à la bordure.
2. Le sous-processus doit avoir un évènement de début ainsi qu'un évènement de fin qui symbolise la continuité du flux du processus de niveau supérieur. En ce sens, l'évènement de départ ne doit pas contenir de marqueur car il ne traduit pas un évènement de type « catch » mais simplement le déclencheur du sous-processus faisant suite à la séquence du processus principal.



Les sous-processus sont donc généralement utilisés lorsqu'il est nécessaire de décrire le comportement d'une activité à un niveau plus fin de détail. Cela peut avoir différents avantages :

- En premier lieu, ils permettent de simplifier le processus afin de pouvoir le visualiser du début à la fin sur une même page, en minimisant le niveau de détail.
- Au-delà de l'aspect visuel, ils ont un véritable avantage méthodologique. Ils permettent de modéliser les processus dans une logique « top-down » c'est-à-dire avec un raffinement progressif du niveau de détail. En fonction des besoins, il n'est d'ailleurs pas obligatoire de modéliser le contenu de chaque sous-processus. Ils peuvent alors rester affichés dans leur format réduit.
- Les sous-processus sont également très utiles lorsque l'exécution de plusieurs tâches peut être influencée par le même évènement. L'occurrence d'un évènement intermédiaire de frontière appliquée à un sous-processus aura un impact sur l'ensemble de la séquence composant le sous-processus.

On pourrait par exemple vouloir modéliser que le départ d'un client durant le sous-processus « préparation de la commande » interrompt le processus principal. On ajoute un évènement de frontière de type signal sur le sous-processus « préparation de la commande ». Peu importe où en est le flux de séquence du sous-processus « préparation de la commande », si le signal du départ du client est capté, alors le sous-processus se termine ainsi que le processus principal.

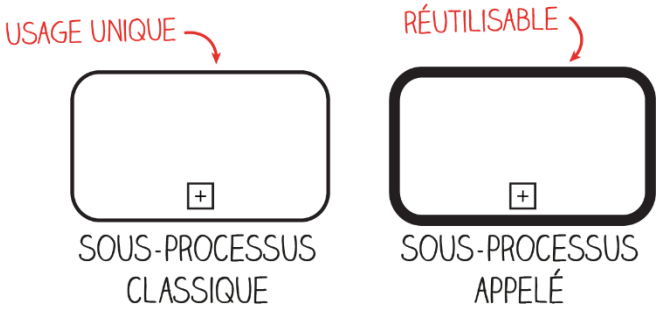


Il existe différents types de sous-processus, que nous allons maintenant présenter.

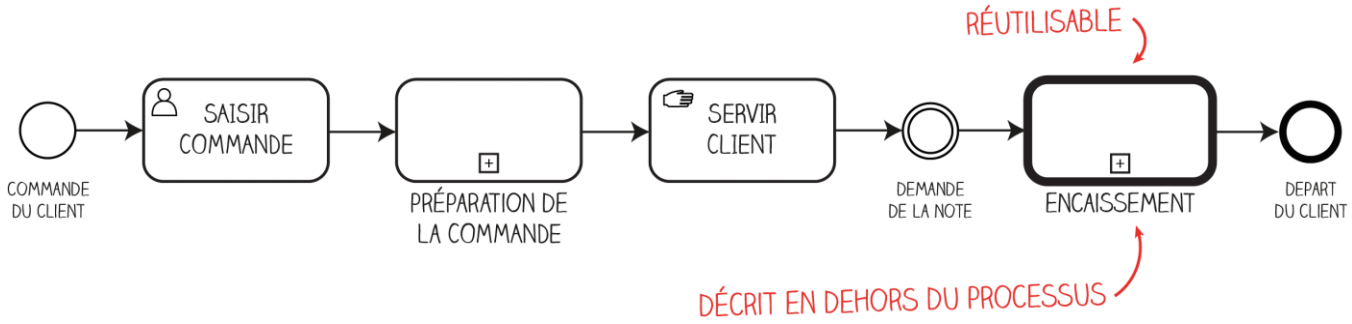
3. LES ACTIVITES REUTILISABLES

Un sous-processus fait partie intégrante du processus principal. Cependant, il est possible qu'un sous-processus puisse être utilisé dans plusieurs processus de niveau supérieur. Les activités ***réutilisables*** ou « ***call activity*** » permettent de ne pas re-modéliser des sous-processus requis dans plusieurs processus.

Le sous-processus est alors indépendant du processus principal. On dit qu'il est « appelé ». On le distingue d'un sous processus classique par une bordure épaisse.



Par exemple, nous pourrions définir un sous-processus « Encaissement » qui pourrait être réutilisé dans notre processus de « service au client » mais également dans le processus « vente à emporter » ou encore « service au bar ». Le sous-processus « Encaissement » serait alors appelé.



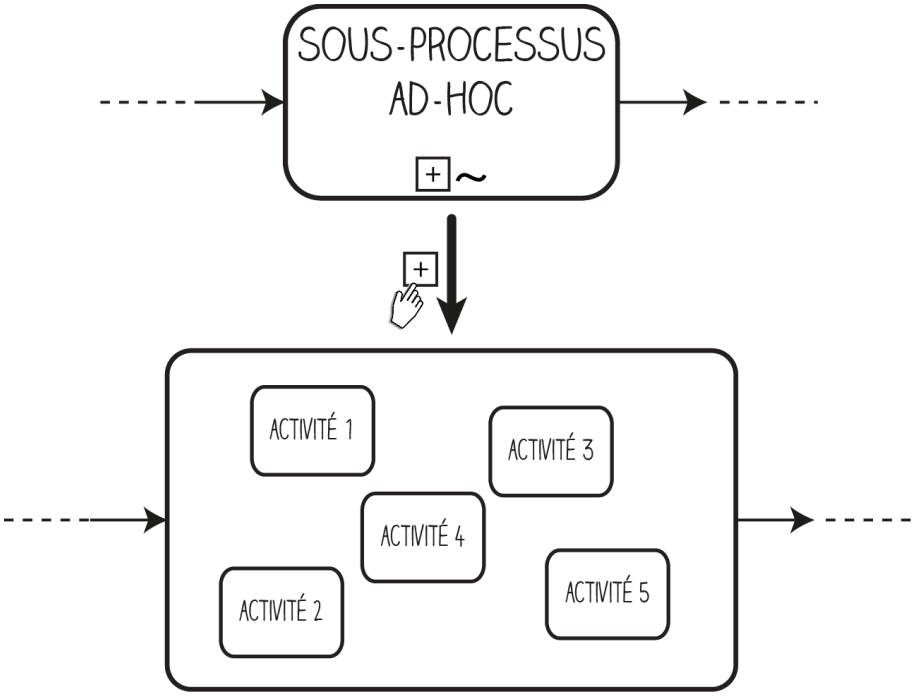
4. LES SOUS-PROCESSUS INFORMELS

Comme toute règle a ses exceptions, il existe des types de sous-processus sans évènement de départ. Il s'agit des sous processus ***parallèle*** et ***ad-hoc***. Ils sont composés d'un ensemble d'activités mais qui ne sont pas interconnectés par un flux de séquence.

Pour le sous-processus ***parallèle***, toutes les activités du sous-processus démarrent en parallèle et peuvent être réalisées dans n'importe quel ordre. Elles doivent cependant être toutes achevées pour terminer le processus.

Le sous-processus ***ad-hoc***, que l'on spécifie avec un marqueur tilde, est une variante dans laquelle toutes les activités n'ont pas besoin d'être réalisées pour terminer le processus. Il se termine lorsque l'acteur exécutant le sous-processus décide qu'il est terminé.

Ces deux types de sous-processus sont très peu utilisés par les modélisateurs souhaitant implémenter leur processus dans un moteur. Mais ils peuvent avoir un intérêt pour modéliser des processus métiers informels ou difficilement séquençables.



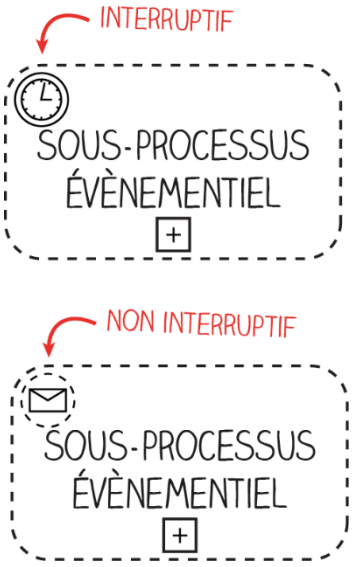
5. LES SOUS-PROCESSUS *EVENEMENTIELS*

Autre exception, les sous-processus ***évènementiels***, spécifiquement conçus pour pouvoir être instanciés grâce à un évènement donné, comme un message ou un timer par exemple.

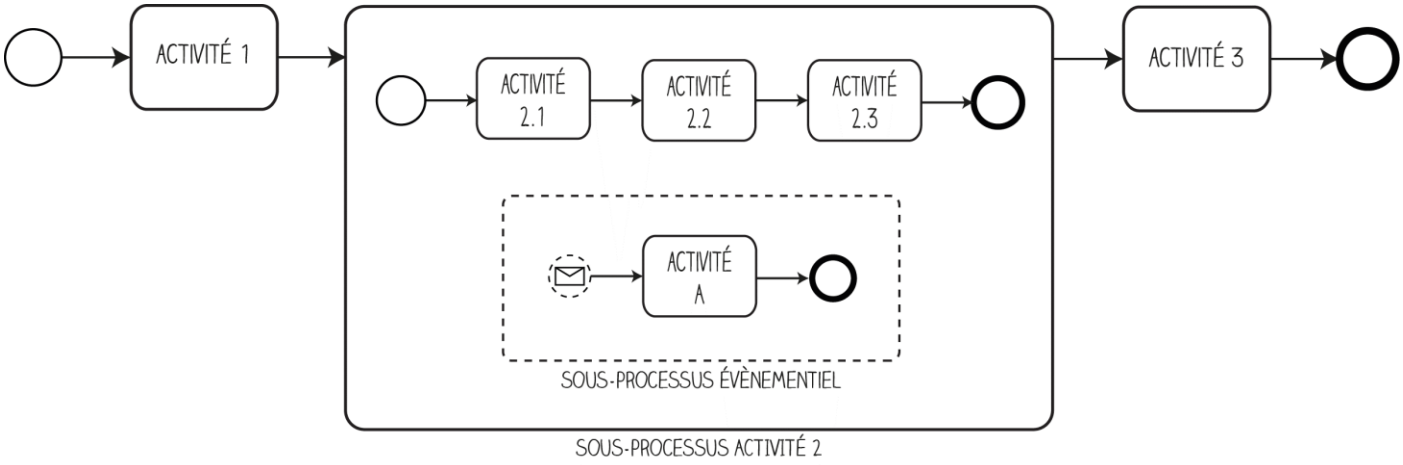
Ainsi, ce type de sous-processus ne fait pas partie de la séquence normale du processus principal ; il n'y a pas de flux entrant et sortant car le sous-processus est déclenché par l'évènement, et non par la suite du flux comme dans un sous-processus classique.

Les sous-processus évènementiels ne peuvent contenir qu'un seul évènement déclencheur dont il faut spécifier la nature par le marqueur correspondant. Durant le processus principal, cet évènement peut ne pas se produire, ou se produire une ou plusieurs fois. Il peut ou pas interrompre le processus principal. Tout comme pour les évènements intermédiaires de frontière, on distingue le caractère interruptif de l'évènement par sa bordure en pointillé ou en trait plein.

On reconnait un sous-processus évènementiel par ses bordures en pointillés. Dans son format réduit, on inscrit dans le coin en haut à gauche un marqueur spécifiant le type d'évènement déclencheur.

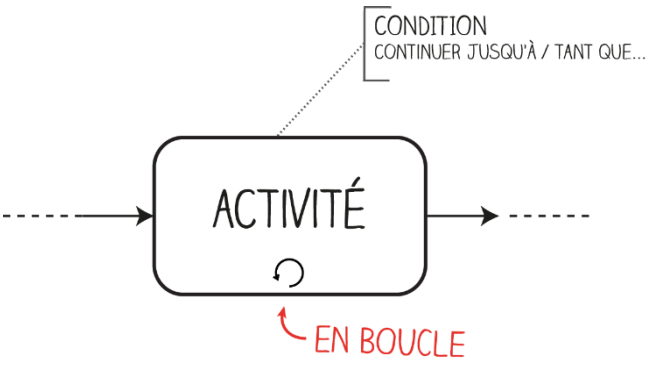


Prenons un exemple générique montrant plusieurs hiérarchies de processus. Notre processus principal est composé de 3 activités. Celle du milieu est un sous-processus, lui-même composé de 3 autres activités. L'évènement de départ correspond à la continuité du flux et ne porte donc pas de marqueur. Durant l'exécution de cette séquence d'activités, il y a un sous-processus évènementiel qui est en attente de réception d'un message. Si nous étendons ce sous-processus, nous voyons que celui-ci est composé d'une unique tâche. Ce sous-processus évènementiel n'interrompt pas l'exécution du processus parent car l'évènement est noté en pointillé. Ainsi, tant que le sous-processus parent n'est pas terminé, le sous-processus évènementiel sera exécuté à chaque réception de message. Cela peut arriver une fois, plusieurs fois ou même aucune fois. Une fois le sous-processus terminé, le flux de séquence reprend sur le processus principal.

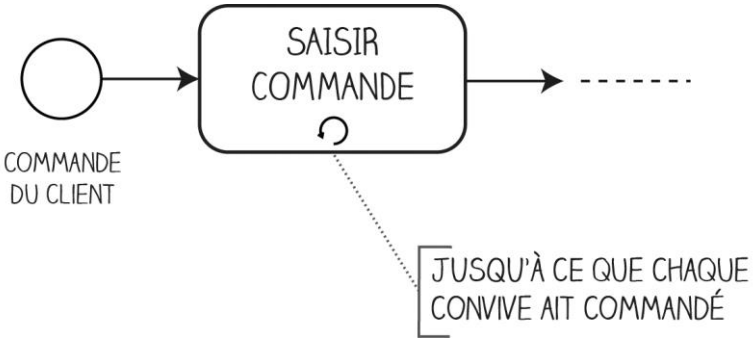


6. LA MULTI-INSTANCES

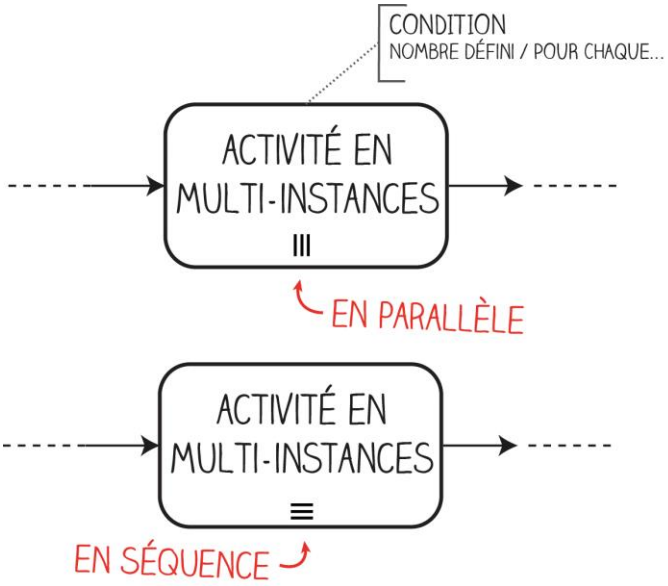
Tâche ou sous-processus peuvent également avoir d'autres types de marqueur indiquant le nombre d'instances de l'activité. Ainsi, le marqueur « ***boucle*** » permet de spécifier qu'une activité va se répéter tant qu'une condition n'est pas remplie. A la fin de l'exécution de l'activité, on vérifie si la condition est vraie. En fonction, on réalise de nouveau l'activité, ou on continue le flux de séquence. Cette condition doit être précisée en annotation de la tâche.



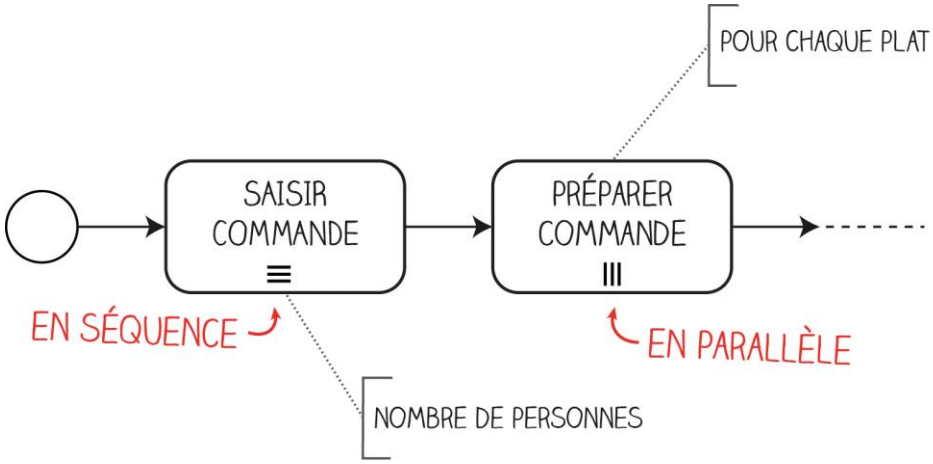
Par exemple, pour la tâche « saisir commande », celle-ci pourra être exécutée jusqu'à ce que toutes les personnes à la table aient commandé. C'est donc une tâche qui boucle en fonction d'une condition indiquée en commentaire. Lorsque tout le monde a commandé, le processus continu.



Il existe deux autres marqueurs pour représenter la multi-instanciation des activités. A la différence de la boucle qui se base sur une condition pour re-exécuter l'activité, ces marqueurs se basent sur un nombre d'instances prédéfini.
Ces instances peuvent s'exécuter en parallèle ; c'est-à-dire en même temps ; le marqueur est alors composé de ***3 traits verticaux***, on appelle cela un ***traitement par lot*** ; ou elles peuvent être réalisées en ***séquence***, c'est-à-dire les unes après les autres ; le marqueur est alors composé de ***3 traits horizontaux***. Une fois l'ensemble des instances réalisées, le flux de séquence continu.



Pour la tâche « Saisir commande », nous pourrions également envisager une exécution en séquence pour chaque personne de la table. Ici, on n'exprime pas une condition mais un nombre d'instances défini par le nombre de personnes. On saisit la commande de chaque convive l'une après l'autre. Contrairement aux différentes tâches de saisie de commandes qui ne peuvent être faites simultanément, les différentes instances de la préparation de la commande concernant chaque personne peuvent être réalisées en parallèle. Ces deux tâches illustrent donc le concept de multi- instances, la première en séquence et la seconde en parallèle.



Nous venons d'appréhender beaucoup de nouvelles notions autour des activités. Il nous est possible maintenant de modéliser de façon précise les différentes activités, tâches et sous-processus, concepts qui sont des éléments essentiels à la description d'un processus. **Ne vous effrayez pas de tous ces détails. Dans la pratique on utilise 20% des concepts dans 80% des cas !**