|  |  |  |
| --- | --- | --- |
| **Sciences de**  **l’Ingénieur** | **Compétences travaillées :**  - Analyser le besoin, l’organisation matérielle et fonctionnelle d’un produit par une démarche d’ingénierie système  - Analyser le traitement de l’information  - Analyser le comportement d’un objet à partir d’une description à événements discrets  - Valider les modèles établis pour décrire le comportement d’un objet  - Traduire le comportement attendu ou observé d’un objet  - Mettre en œuvre une simulation numérique à partir d’un modèle multi-physique pour qualifier et quantifier les performances d’un objet réel ou imaginé | **Connaissances associées :**  - Outils d’ingénierie-système : diagrammes fonctionnels, définition des exigences et des critères associés, cas d’utilisations, analyse structurelle  - Algorithme, programme, langage informatique  - Diagramme états-transitions, algorithme  - Analyse des écarts de performances  - Structures algorithmiques (variables, fonctions, structures séquentielles, itératives, répétitives, conditionnelles), diagramme d’états-transitions  - Paramètres de simulation : durée, incrément temporel, choix des grandeurs affichées, échelles adaptées à l’amplitude et la dynamique des grandeurs simulées |
| **Durée :** 2 h |
| Référence :  Hydrao\_Solution\_1 |
| **Matériel :**  Système didactique HYDRAO DIDACT |
| **Logiciel :**  MatLab Stateflow 2019b pour la simulation |
| **Documents :**  Dossier technique du système |
| **Pré-requis :**  - Premiers éléments sur les diagrammes SysML  - Premiers éléments sur les machines à états | |

# Problème scientifique et technologique :

Le système de douche intelligent imaginé par Hydrao pour réduire le volume d’eau consommé lors d’une douche, et conséquemment l’énergie nécessaire pour chauffer cette eau, repose à la fois sur une limitation du débit d’eau, et l’implication de l’utilisateur en l’informant de ses consommations.

On se propose dans cette première activité (Hydrao\_Solution\_1) de prendre en main ce système du point de vue utilisateur, puis d’en étudier plusieurs aspects fonctionnels au niveau système. La description du fonctionnement est basée sur les diagrammes états-transitions rencontrés dans SysML.

Une deuxième activité (Hydrao\_Solution\_2) permettra d’étudier plusieurs aspects des choix technologiques toujours au niveau système.

Enfin une troisième activité (Hydrao\_Solution\_3) permettra de valider le concept par prototypage rapide.

# Travail à effectuer :

Nota : il conviendra de réaliser un dossier réponse documenté à l’aide de copies d’écran pertinentes si besoin.

# Présentation générale du système

Le système de douche intelligent d’Hydrao est composé d’un pommeau de douche (« Aloé » dans notre cas) autoalimenté, et connecté à un smartphone où l’application « HYDRAO SMART SHOWER » est installée : <https://www.hydrao.com/fr/>.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| RÃ©sultat de recherche d'images pour "Hydrao" |  |  |  |  |

Principe de fonctionnement :

* la couleur des LED éclairant le jet d’eau change avec le volume d’eau écoulé, informant l’utilisateur de sa consommation d’eau en temps réel ;
* grâce à une application installée dans un smartphone, l’utilisateur autorisé peut consulter l’historique des consommations, personnaliser et paramétrer certains éléments du système ;
* les données anonymisées sont transmises à un cloud, permettant des analyses statistiques ;
* une turbine-génératrice assure l’auto-alimentation électrique du pommeau ; le pommeau ne comporte ni pile ni batterie.

## 1ère partie : prise en main du pommeau et de l’application en tant qu’utilisateur

Vous disposez des documents : guide de démarrage rapide, fiche produit, notice du limiteur de débit (fichiers : Guide\_demarrage\_rapide.pdf, Fiche\_Produit\_HYDRAO\_Aloé.pdf, et HYDRAO\_Notice\_limiteur\_debit.pdf), et du système didactique HYDRAO DIDACT. Si besoin vous pouvez aussi consulter la faq sur <https://www.hydrao.com/fr/assistance/faq>.

**Question 1 : installation et prise en main du pommeau « Aloé »**

- Comparez l’installation du pommeau « Aloé » avec l’installation d’un pommeau classique.

- Est-il utilisable sans l’application « HYDRAO SMART SHOWER » ?

- Quel est le processus d’installation et de mise en fonctionnement de l’application « HYDRAO SMART SHOWER » ?

- Que permet cette application ?

- Exposez succinctement les axes visés par Hydrao d’une expérience utilisateur du système de douche intelligent.

## 2ème partie : comportement attendu pour l’utilisateur

Vous disposez des diagrammes SysML et du schéma synoptique du pommeau Aloé (fichier « Hydrao\_Aloe\_SysML.pdf »). Si besoin, des informations sur les diagrammes des cas d’utilisation et les diagrammes états-transitions figurent dans le document ressource « éléments sur quelques diagrammes SysML » (fichier « Ressource\_Elements\_sur\_Diagrammes\_SysML.pdf »).

**Question 2 : diagramme des cas d’utilisation (use case diagram, ou UC)**

-2.1- A partir du diagramme des cas d’utilisation, listez les acteurs principaux et les acteurs secondaires. Que signifient acteur principal et acteur secondaire ?

-2.2- Quelle est la signification de la flèche entre les deux acteurs « Utilisateur » et « Utilisateur ayant les droits sur l'appareil connecté où est installée l'application » ? Quelle en est la conséquence ?

-2.3- Indiquez dans le tableau ci-dessous les cas d’utilisation accessibles par chacun de ces deux acteurs.

|  |  |  |
| --- | --- | --- |
|  | Utilisateur | Utilisateur ayant les droits sur l’appareil connecté où est installée l’application |
| se doucher |  |  |
| --------> être informé en temps réel visuellement du volume d’eau écoulé depuis le début de la douche |  |  |
| consulter l’historique des consommations |  |  |
| personnaliser et paramétrer : seuils et couleurs, douche de référence, coûts unitaires de l’eau et de chauffage |  |  |

-2.4- Le cas d’utilisation « être informé en temps réel visuellement du volume d’eau écoulé depuis le début de la douche » est relié au cas « se doucher » par une liaison stéréotypée « include ». Compte tenu de cette relation, quel est le comportement attendu du système ?

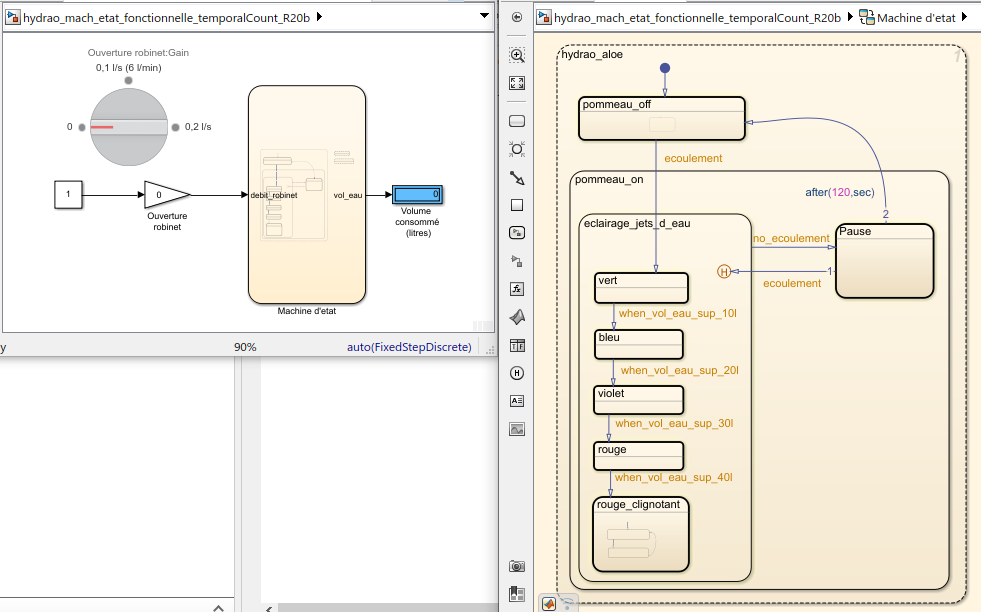
**Dans la suite, on s’intéresse principalement au cas d’utilisation « être informé en temps réel visuellement du volume d’eau écoulé depuis le début de la douche ».**

**Question 3 : diagramme états-transitions (statechart diagram) – comportement attendu choix technologiques non encore faits**

Le diagramme états-transitions SysML intitulé « machine d’état fonctionnelle » décrit le comportement attendu du cas d’utilisation « être informé en temps réel visuellement du volume d’eau écoulé depuis le début de la douche », les choix technologiques n’étant pas encore faits.

Pour simuler le comportement attendu décrit par le diagramme états-transitions SysML, exécutez la simulation sous Matlab Simulink Stateflow de la « machine à états fonctionnelle » (fichier : « ***hydrao\_mach\_etat\_fonctionnelle\_temporalCount\_Rxxx.slx*** »), avec ***Rxxx*** désignant la version de matlab).

Le bouton rotatif « Ouverture robinet : Gain » permet de simuler l’écoulement de l’eau à travers le pommeau en fixant le débit parmi trois valeurs : 0 l/min (arrêt), 5 l/min et 10 l/min. Lorsque la simulation a démarré, vous pouvez simuler une douche en ouvrant et fermant le robinet.



A partir du diagramme états-transitions SysML, et de la simulation pour confirmer votre interprétation, répondez aux questions suivantes.

-3.1- Quel est le rôle du pseudo-état « H\* » ? En supposant que l’état composite « eclairage\_jet\_d\_eau » soit actif et l’état « bleu » actif, décrivez l’évolution de la machine lors de la fermeture du robinet puis de sa réouverture une minute après.

La configuration des états hiérarchisés actifs à un instant donné de l’exécution de la machine à états est appelée une **configuration d’état actif**.

On désigne ici par : diagramme des configurations d’états actifs, le graphe de toutes les configurations d’états actifs accessibles.

-3.2- Complétez le diagramme des configurations d’états actifs ci-dessous (Etats actifs dans le graphe). Pour chacune de ces configurations, indiquez aussi les états hiérarchisés tracés par le pseudo-état historique (Historique dans le graphe). Ce diagramme décrit-il correctement le comportement attendu ?

Diagramme des configurations d’états actifs :



-3.3- En s’inspirant de l’analyse précédente et en introduisant une variable (notée « histo » dans le diagramme) traçant une configuration d’état actif, mais sans pseudo-état historique, complétez les diagrammes états-transitions SysML ci-dessous répondant au comportement attendu.

Diagramme avec états composites :



Diagramme sans état composite (hormis « rouge\_clignotant ») :



-3.4- Une solution, sans variable traçant une configuration d’état actif et sans pseudo-état historique, consiste à utiliser des gardes dont les conditions portent sur « vol\_eau ». Complétez les diagrammes états-transitions SysML ci-dessous répondant au comportement attendu.

Diagramme avec états composites :



Diagramme sans état composite (hormis « rouge\_clignotant ») :



-3.5- Enfin, à partir des analyses précédentes, sans variable traçant une configuration d’état actif, sans pseudo-état historique, sans garde, et sans état composite (hormis rouge\_clignotant), dessinez un diagramme états-transitions SysML répondant au comportement attendu.

Pour cela, déterminez préalablement tous les couples possibles (état actif, état mémorisé dans l’historique) ou ce qui revient au même (état actif, valeur prise par la variable « histo »).

## 3ème partie : ce qu’il faut retenir

**A) Cas d’utilisation (use case – uc) SysML :** les cas d’utilisation expriment les services rendus par le système aux différents acteurs.

|  |  |
| --- | --- |
| **Acteurs :** | |
|  | Un **acteur** est qualifié de **principal** pour un **cas d'utilisation** (inscrit dans un ovale) lorsque ce cas lui rend un service.  • Le système, délimité par sa **frontière**, est développé pour rendre des services aux acteurs principaux.  • Les autres acteurs sont qualifiés de **secondaires**. Ils contribuent à la réalisation des services.  • Chaque cas d’utilisation est relié par une ligne aux acteurs impliqués. |
|  | **Généralisation / spécialisation entre acteurs :**  • L’acteur B est une généralisation de l’acteur A.  • Autrement dit, A est une spécialisation de B.  • A hérite des cas d’utilisation de B (tous les cas d'utilisation accessibles à B le sont aussi à A, mais l'inverse n'est pas vrai). |
| **Relations standards entre cas d’utilisation :** | |
|  | **Inclusion :**  • Le cas A inclut le cas B.  • Autrement dit, B est une partie de A.  • B est toujours exécuté au cours de l’exécution de A. |
|  | **Extension :**  • Le cas A étend le cas B.  • A est exécuté lors de l’exécution de B si une condition est remplie.  • L’exécution de A est donc optionnelle. |
|  | **Généralisation / spécialisation entre cas d’utilisation :**  • Le cas B est une généralisation du cas A.  • Autrement dit, A est une spécialisation (cas particulier) de B.  • A hérite des acteurs de B. |

**B) Principaux concepts sur les diagrammes états-transitions (SysML)**

**Machine à états (state machine) – Diagramme états-transitions (statechart)**

• Une **machine à états** ou **machine d’état** (ou **automate**) permet de modéliser le comportement d’un **système à événements discrets** (SED).

• Un **diagramme états-transitions (statechart)** décrit graphiquement une machine à états.

• La modélisation par un diagramme états-transitions du comportement d’un SED s’effectue en décrivant :

* les états possibles du système ;
* les transitions, entre états ou internes aux états, déclenchées par des occurrences d’événements sous certaines conditions ;
* les comportements associés aux états et aux transitions.

|  |  |
| --- | --- |
|  | **Etat simple (simple state) :** un état est soit actif, soit inactif.  • Un état devient actif dès que le système entre dans l’état, et inactif dès que le système en est sorti (ou l’a quitté)  • Un état simple est décrit par les compartiments facultatifs : nom, comportements internes, et transitions internes.  • Comportement (ou effet) interne :   * entry/ : effectué lors de l’entrée dans l’état ; * do/ : effectué durant l’état actif (après les comportements d’entrée et avant les comportements de sortie), effectué tant que l’état est actif ou jusqu’à ce qu’il se termine par lui même ; * exit/ : effectué lors de la sortie de l’état.   Les comportements d’entrée « entry\_behavior » et de sortie « exit\_behavior » sont de « courtes » durées et ne peuvent pas être interrompus.  Les comportements non terminés durant l’état actif « doActivity\_behavior » sont interrompus en cas de sortie de l’état.  • Transition interne (écriture habituelle d’une transition). Une transition interne s’exécute sans quitter l’état dans lequel elle est définie, ni y rentrer ; les comportements d’entrée et de sortie ne sont donc pas effectués. |
|  | **Etat composite (composite state) :** un état composite est décrit par les compartiments : nom, comportements internes, transitions internes, auxquels s’ajoute le compartiment de décomposition.  • Les lignes pointillées divisent le compartiment de décomposition en **régions** dites **orthogonales** qui concourent ensemble au comportement. Elles sont aussi dites concurrentes  • Un état composite est un état contenant une ou plusieurs régions, et chaque région contient un graphe éventuellement hiérarchique.  • L’entrée dans un état composite provoque l’activation de toutes les régions (après avoir effectué le comportement d’entrée).  • L’entrée dans un état composite et les comportements associés sont effectués séquentiellement du niveau hiérarchique le plus élevé au niveau le plus faible de chaque région.  • La sortie de l’état composite provoque la désactivation de toutes les régions et de tous les états actifs.  • La sortie d'un état composite et les comportements associés sont effectués séquentiellement du niveau hiérarchique le plus faible au niveau le plus élevé de chaque région, pour finir par la sortie de l’état composite lui-même. |
|  | **Etat composite (composite state) :** un état composite peut être affiché avec une décomposition cachée. |
|  | **Etat final (final state) :** un état final marque la fin d’activation d’une région.  • L’état final est (potentiellement) stable, comme tout état. |
|  | **Pseudo-état initial (initial pseudo state) :** un pseudo-état initial cible l’état actif par défaut d’une région.  • La transition sortante du pseudo-état initial peut avoir un comportement, mais pas de déclencheur ou de garde.  • Contrairement à l’état final, le pseudo-état initial est transitoire. |
|  | **Pseudo-état de jonction (junction pseudo state) :** un pseudo-état de jonction réalise une branche conditionnelle statique : toutes les conditions de garde sont évaluées avant l'exécution de toute transition composée contenant ce pseudo-état.  • Si plus d'une transition composée est validée (plusieurs chemins possibles), l'une d'entre elle est choisie mais le choix n'est pas défini. Il est très conseillé de les rendre exclusives.  • Une garde prédéfinie notée « else » peut être définie à au plus une transition sortante.  • Seuls les segments entrants peuvent contenir des déclencheurs. |
|  | **Pseudo-état de choix (choice pseudo state) :** un pseudo-état de choix réalise une branche conditionnelle dynamique : les gardes situées après le pseudo-état de choix sont évaluées au moment où il est atteint.  • Si plus d'une garde est évaluée à vrai, l'une des transitions correspondantes est sélectionnée, mais le choix n'est pas défini.  • Si aucune garde des segments sortants n’est évaluée à vrai, alors le modèle est considéré comme mal formé. Pour éviter cela, il est conseillé d’utiliser une transition sortante avec la garde « else » prédéfinie.  • Il possède un seul segment entrant et deux ou plusieurs segments sortants. |
|  | **Pseudo-état fourche (fork pseudostate) :** un pseudo-état fourche divise une transition entrante en plusieurs transitions sortantes se terminant sur des sommets des régions orthogonales d’un état composite.  • Les transitions sortant d’une fourche ne peuvent pas avoir de garde ou de déclencheur.  **Pseudo-état jointure (join pseudostate) :** un pseudo-état jointure fusionne plusieurs transitions provenant de sommets de régions orthogonales ; il effectue une synchronisation.  • Les transitions entrant sur une jointure ne peuvent pas avoir de garde ou de déclencheur.  • Toutes les transitions entrantes doivent se terminer avant que l'exécution puisse se poursuivre via une transition sortante. |
|  | **Pseudo-état historique peu profond (shallow history pseudo state), ou superficiel :** un pseudo-état historique peu profond trace l’état actif le plus récent de sa région (mais pas les éventuels états actifs internes à cet état).  • Une transition entrant dans le pseudo-état historique peu profond équivaut à une transition entrant dans le dernier état actif tracé par l’historique ; le comportement d’entrée de cet état est effectué.  • Une transition (optionnelle) du pseudo-état historique peu profond vers un état par défaut permet de préciser le cas où l’état composite n’a jamais été actif auparavant. En l’absence de cette transition, c’est l’état actif par défaut de la région qui est activé. |
|  | **Pseudo-état historique profond (deep history pseudo state) :** un pseudo-état historique profond trace la configuration d’état actif la plus récente de sa région.  • Une transition entrant dans le pseudo-état historique profond équivaut à une transition entrant dans les états les plus internes de la dernière configuration d’état actif tracée par l’historique ; les comportements d’entrée rencontrés en parcourant les chemins de restauration de la configuration d’état actif sont effectués.  • Une transition (optionnelle) du pseudo-état historique profond vers un état par défaut permet de préciser le cas où l’état composite n’a jamais été actif auparavant. |
|  | **Pseudo-état de terminaison (terminate pseudostate) :** le pseudo-état de terminaison marque la fin de l’exécution d’une machine à états.  • L’exécution de la machine à états prend fin immédiatement lors de l’entrée dans un pseudo-état de terminaison.  • La machine à états ne quitte aucun état et n'effectue aucun comportement de sortie. Tout comportement en cours d'exécution est automatiquement abandonné. |
|  | **Evénement (event) :** seule l’occurrence (l’apparition) d’un événement peut déclencher une transition.  **Types d’événements SysML :**  - signal (signal event) : désigné par son nom  - changement (change event) : when(boolean\_expression)  - temporel (time event) : after(temporal\_duration), at(date),  when(date = date0)  - d'appel (call event) : appel d’une opération désigné par son nom  **Notations hors SysML :** l’événement front montant de , noté , correspond au passage de la valeur 0 (resp. faux) à la valeur 1 (resp. vrai) de la variable binaire (resp. booléenne) .  • Si a et b sont deux variables indépendantes alors : |
|  | **Transition :** une transition est un arc orienté qui relie un sommet dit source à un sommet dit cible.  • Une transition est définie par l’expression BNF :  [<trigger> [‘,’ <trigger>]\* [‘[‘ <guard>’]’] [‘/’ <behavior-expression>]]  où :  <trigger> est un déclencheur lié à un événement ;  <guard> est une expression booléenne (garde) ;  <behavior-expression> est une expression spécifiant un comportement.  • Une transition est dite :  - atteinte, lorsque l'état source est actif ;  - traversée (ou franchie ou exécutée) lorsqu'elle est en cours ;  - terminée, lorsque l’état cible est actif.  **Condition de garde (guard)**  • L’expression booléenne de la condition de garde associée à une transition est évaluée seulement lors de l’occurrence de l’événement.  • En l’absence d’expression booléenne explicite, le résultat de la condition de garde est vrai.  • Si le résultat de cette condition est faux alors la transition n’est pas déclenchée et l’occurrence de l’événement est perdue.  • Dans une transition simple, la garde est évaluée avant le déclenchement de la transition.  • Dans les transitions composées impliquant plusieurs gardes, toutes les gardes sont évaluées avant qu’une transition ne soit déclenchée, à moins qu’il y ait des points de choix le long d’un ou plusieurs des chemins.  • La syntaxe d’une condition de garde vérifiant si un état nommé state0 est actif est : [in state0]. Celle vérifiant qu’il n’est pas actif est : [not in state0].  **Comportement (behavior)**  • Le comportement associé à la transition est effectué lors de l’exécution de la transition. |
|  | **Transition d’achèvement (completion transition) :** une transition d’achèvement est dépourvue de déclencheur explicite.  • Un événement d’achèvement est implicitement associé à une transition d’achèvement et peut la déclencher.  • Pour un état simple, un événement d’achèvement apparait automatiquement à la fin des comportements associés à l’état source d’une transition d’achèvement.  • Pour un état composite, un événement d'achèvement est généré si toutes les activités internes (exit/, do/, …) ont terminé leur exécution, et toutes ses régions orthogonales ont atteint un état final. |

**Etat (state) vs pseudo-état (pseudostate)**

Un **état** correspond à une situation « durable » du système, tandis qu’un **pseudo-état** correspond à une situation « transitoire » du système.

**Sommet (vertex / vertices)**

• Un **sommet** désigne différents types de nœuds du graphe : états, pseudo-états, points de connexion.

**Règle d’évolution d’une transition simple entre deux états simples**

• Dans ce cas, une **transition** est dite **validée** (ou **éligible**, ou autorisée, ou permise) si et seulement si :

* son état source est actif ;
* l’événement occurrent (qui apparait) est un déclencheur (trigger) de la transition ;
* le résultat de la condition de garde associée à ce déclencheur est vrai.

• Plusieurs transitions peuvent être validées par la même occurrence d'événement. Ainsi, la validation est une condition nécessaire mais non suffisante pour déclencher (ou franchir) une transition.

• A l’occurrence d’un événement, la machine à états :

1. détermine les transitions validées (voir ci-dessus) ;
2. choisit la transition à déclencher le cas échéant ;
3. arrête le comportement durant l’état actif (do/) de l’état source le cas échéant ;
4. effectue le comportement de sortie (exit/) de l’état source le cas échéant ;
5. désactive l’état source ;
6. effectue le comportement associé à la transition le cas échéant ;
7. active l’état cible ;
8. effectue le comportement d’entrée (entry/) de l’état cible le cas échéant ;
9. démarre le comportement durant l’état actif (do/) de l’état cible le cas échéant ;
10. attend une nouvelle occurrence d’un événement.

• Si aucune transition n’est validée alors l’occurrence de l’événement est perdue.

• Le choix de la transition à déclencher parmi les transitions validées résulte d’un éventuel ordre de priorité ou « du hasard ». Il convient bien sûr d’éviter cette circonstance notamment par le choix des conditions des gardes.

• Une occurrence d’événement permet de déclencher une seule transition. En conséquence, si au moins deux transitions étaient validées alors une seule serait déclenchée. Cela est vrai sauf dans le cas de régions orthogonales où une occurrence permet de déclencher une transition par région orthogonale.

**Configuration d’état (state configuration) ou « état global » – Configuration d’état actif**

• Une **configuration d'état** est un ensemble d’états hiérarchisés acceptable d’un diagramme.

• Une **configuration d'état actif** est l’ensemble des états hiérarchisés actifs à un moment de l’exécution de la machine à états.

• Une machine à états évolue d’une configuration d’état actif à une autre en réponse à des occurrences d'événements.

• Une **configuration d'état actif** est dite **stable** lorsque :

* aucune autre transition de cette configuration n'est validée et,
* tous les éventuels comportements d'entrée de cette configuration sont terminés (les éventuels comportements en cours d’état actifs (do/) de cette configuration peuvent continuer à s’exécuter).

## 4ème partie : exercices

### Exercice 1 : transition vers un état composite

Le diagramme états-transitions SysML intitulé « machine d’état fonctionnelle » (fichier Hydrao\_Aloe\_SysML.pdf) est modifié comme suit : le pseudo-état historique est supprimé et la transition étiquetée par « ecoulement » de l’état « pause » vers le pseudo-état historique est remplacée par une transition étiquetée par « ecoulement » de l’état « pause » vers l’état composite « eclairage\_jets\_d\_eau ». Dessinez le diagramme des configurations d’états actifs. Concluez.

### Exercice 2 : modification du comportement au retour de l’état « pause »

On souhaite modifier le comportement de la « Machine d’état fonctionnelle » de la façon suivante : lors du retour de l’état « pause », la séquence d’éclairage des couleurs déjà effectuées est refaite telle que chaque couleur est allumée durant une seconde. Par exemple, supposons que l’éclairage était violet avant d’entrer dans l’état « pause », lors du retour de l’état « pause », la séquence d’éclairage serait : vert pendant une seconde, bleu pendant une seconde, puis violet jusqu’à l’événement « when(vol\_eau>30 l) ». Modifiez en conséquence le diagramme états-transitions SysML de la « Machine d’état fonctionnelle ».

### Exercice 3 : ajout d’une séquence de couleurs de début de douche

Hydrao a ajouté une séquence de couleurs de début de douche : bleu, blanc, rouge, chaque couleur durant environ une seconde. En cas d’arrêt d’écoulement d’eau pendant cette séquence de début de douche, puis d’une reprise de l’écoulement, la séquence de couleurs de début de douche est entièrement recommencée. Modifiez en conséquence le diagramme états-transitions SysML de la « Machine d’état fonctionnelle ».

### Exercice 4 : information visuelle contre les risques de brûlures

Pour éviter les risques de brûlures (douche donnée à une autre personne, personne insensible à la chaleur), on souhaite informer visuellement les utilisateurs d’une eau trop chaude en éclairant les jets d’eau en jaune clignotant (couleur du soleil).

Cette information visuelle sera prioritaire sur celle du volume d’eau écoulé. L’information visuelle sur le volume d’eau écoulé devra être rétablie lorsque la température de l’eau sera redevenue conforme.

On supposera qu’un capteur est utilisé pour mesurer la température de l’eau.

a) Ajoutez le cas d’utilisation « être informé en temps réel visuellement d’une eau trop chaude » au diagramme des cas d’utilisation.

b) Modifiez en conséquence le diagramme états-transitions SysML de la « Machine d’état fonctionnelle » en utilisant des états composites.