



ENTERPRISE ARCHITECT

Série de Guides d'Utilisateur

Simulations Dynamiques

Author: Sparx Systems

Date: 7/11/2024

Version: 17.0

CRÉÉ AVEC  **ENTERPRISE
ARCHITECT**

Table des Matières

Simulations Dynamiques	3
À quoi ça ressemble	6
Fenêtres de Simulation	7
Configurer Script de Simulation	10
Activer Script de Simulation	12
Exécuter Simulation de Modèle	13
Simulation Points d'Arrêt	16
Objets et Instances dans Simulation	18
Créer Objets dans une Simulation	19
Détruire Objets dans une Simulation	22
Simulation dynamique avec JavaScript	24
Comportements d'Appels	27
Condition d'Opérande d'Interaction et Comportement du Message	29
Gardes et Effets	31
Déclencheurs	33
Comportement d'Action par Type	35
Simulation d'Activité Structurée	37
Simulation de Valeur de Retour d'Activité	39
Fenêtre Simulation Événements	42
Déclencheurs en Attente	45
Déclencheurs de Re-Signal	46
Multi-filetage - Fourches et Jointures	47
Paramètres Déclencheur	48
Ensemble Déclencheur et Auto-Tir	50
Utiliser Ensembles Déclencheur pour simuler une Séquence d'événements	53
Multi-filetage - Régions State concurrentes	54
Utilisant Diagrammes composites	55
Simulation d'Interface Utilisateur Win32	57
Contrôles UI Win32 pris en charge	59
Win32 Control Valeur Étiquetés	70
Simulation BPMN	71
Créer un Modèle de Simulation BPMN	72
Initialiser Variables et Conditions	74
Comparaison des activités UML et Processus BPMN	76

Simulations Dynamiques

Simulation de Modèle donne vie à vos modèles comportementaux grâce à une exécution instantanée et en temps réel. Associé à des outils de gestion déclencheurs, des événements, des gardes, des effets, des points d'arrêt et des variables Simulation, ainsi qu'à la possibilité de suivre visuellement l'exécution au moment de l'exécution, le simulateur est un moyen polyvalent de « regarder les roues tourner » et de vérifier l'exactitude de vos modèles comportementaux. Avec Simulation vous pouvez explorer et tester le comportement dynamique des modèles. Dans les éditions Corporate, Unified et Ultimate, vous pouvez également utiliser JavaScript comme langage d'exécution au moment de l'exécution pour évaluer les gardes, les effets et d'autres éléments de comportement scriptables.

support étendue des déclencheurs, des ensembles déclencheur, des états imbriqués, de la concurrence, des effets dynamiques et d'autres fonctionnalités Simulation avancées fournit un environnement remarquable dans lequel créer des modèles interactifs et fonctionnels qui aident à explorer, tester et tracer visuellement le comportement complexe des entreprises, des logiciels et des systèmes. Avec JavaScript activé, il est également possible de créer des objets COM intégrés qui effectueront le travail d'évaluation des gardes et d'exécution des effets, ce qui permet à la Simulation d'être liée à un ensemble beaucoup plus vaste de processus dépendants. Par exemple, un objet COM évaluant une condition de garde sur une transition State peut interroger un processus exécuté localement, lire et utiliser un ensemble de données de test, ou même se connecter à un service Web SOA pour obtenir des informations actuelles.

Comme Enterprise Architect utilise un mécanisme Simulation dynamique piloté par script qui analyse et utilise directement les constructions UML, il n'est pas nécessaire de générer du code intermédiaire ou de compiler des « exécutables » de simulation avant d'exécuter une Simulation. Il en résulte un environnement Simulation très rapide et dynamique dans lequel des modifications peuvent être apportées et testées rapidement. Il est même possible de mettre à jour les variables Simulation en temps réel à l'aide de la fenêtre Console Simulation. Cela est utile pour tester des branches et des conditions alternatives « à la volée », soit à un point d'arrêt Simulation défini, soit lorsque la Simulation atteint un point de stabilité (par exemple, lorsque la Simulation est « bloquée »).

Dans l'édition Professional d' Enterprise Architect, vous pouvez parcourir manuellement les simulations (bien qu'aucun JavaScript ne soit exécuté), de sorte que tous les choix sont des décisions manuelles. Cela est utile pour tester le flux d'un modèle comportemental et mettre en évidence les choix et les chemins de traitement possibles. Dans les éditions Corporate, Unified et Ultimate il est possible de :

- Exécutez dynamiquement vos modèles comportementaux
- Évaluer les protections et les effets écrits en JavaScript standard
- Définir et déclencher déclencheurs dans les simulations en cours d'exécution
- Définir et utiliser des ensembles de déclencheurs pour simuler différentes séquences d'événements
- Ensembles déclencheur à déclenchement automatique pour simuler des historiques d'événements complexes sans intervention de l'utilisateur
- Mettre à jour les variables Simulation « à la volée » pour modifier le déroulement des simulations
- Créez et appelez des objets COM pendant une Simulation pour étendre la portée et les possibilités d'entrée/sortie de Simulation
- Inspecter les variables Simulation au moment de l'exécution
- Définir un « prologue » de script pour définir des variables, des constantes et des fonctions avant l'exécution
- Utilisez plusieurs Scripts d'Analyseur avec différents « prologues » pour exécuter la Simulation dans un large éventail de conditions

Dans les éditions Unified et Ultimate, il est également possible de simuler des modèles BPMN.

En utilisant le simulateur Modèle, vous pouvez simuler l'exécution de modèles conceptuels contenant un comportement. Lorsque vous démarrez une Simulation, le modèle actuel Paquetage est analysé et un processus Simulation dynamique est déclenché pour exécuter le modèle.

Pour démarrer avec Simulation, les seules étapes requises sont :

- Construire un diagramme comportemental (State ou Activité pour exécution manuelle ou dynamique, Séquence pour interaction manuelle uniquement)
- Facultatif : charger la disposition « Espace de travail Simulation » - un moyen rapide d'afficher toutes les fenêtres Simulation fréquemment utilisées

- Cliquez sur le bouton Jouer au simulateur

Si le diagramme contient des éléments externes (ceux qui ne sont pas dans le même Paquetage que le diagramme), vous devrez créer un connecteur d'importation du Paquetage du diagramme vers le Paquetage contenant les éléments externes. Pour cela, faites glisser les deux Paquetages de la fenêtre Navigateur sur un diagramme , puis utilisez la flèche Quick Linker pour créer le connecteur entre eux.

Présentation Simulation

Aspect
Présentation du simulateur Modèle
Utilisation de la fenêtre Simulation et Windows associées, et exécution d'une Simulation
Configurer une Simulation et activer un script Simulation
Configurer et utiliser Points d'Arrêt Simulation
Simuler l'utilisation des objets
L'utilisation de différents types d' Action dans Simulation
Effectuer Simulation dynamique avec JavaScript
L'utilisation des Gardes et Effets dans les simulations
L'utilisation des Déclencheurs dans les simulations
Comportements d'Appels et variables
Simulation des retours d'activité
Simulation du comportement d'une activité structurée
Simulation de Processus multithread
Simulation de sous-processus dans Diagrammes séparés
Réalisation de simulations BPMN
Simuler le comportement Dialogue Win32

Plateformes et éditions disponibles

Plateforme/Édition	Détails
Modèles et plateformes pris	Le Modèle Simulator prend actuellement supporte l'exécution des modèles d'activité, d'interaction et Statemachine UML et Processus Métier BPMN sur les

en charge	plateformes Simulation : <ul style="list-style-type: none">• UML de base• BPMN
Support des éditions	Simulation de Modèle est disponible à différents niveaux dans la gamme des éditions d' Enterprise Architect : <ul style="list-style-type: none">• Professional - Simulation manuelle uniquement• Corporate et supérieur - Ajoute une évaluation JavaScript dynamique ; JavaScript est actuellement activé pour Statemachines et les graphiques d'activité ; il n'est pas activé pour diagrammes d'interaction• Unified et Ultimate - Ajoute Simulation BPMN

À quoi ça ressemble

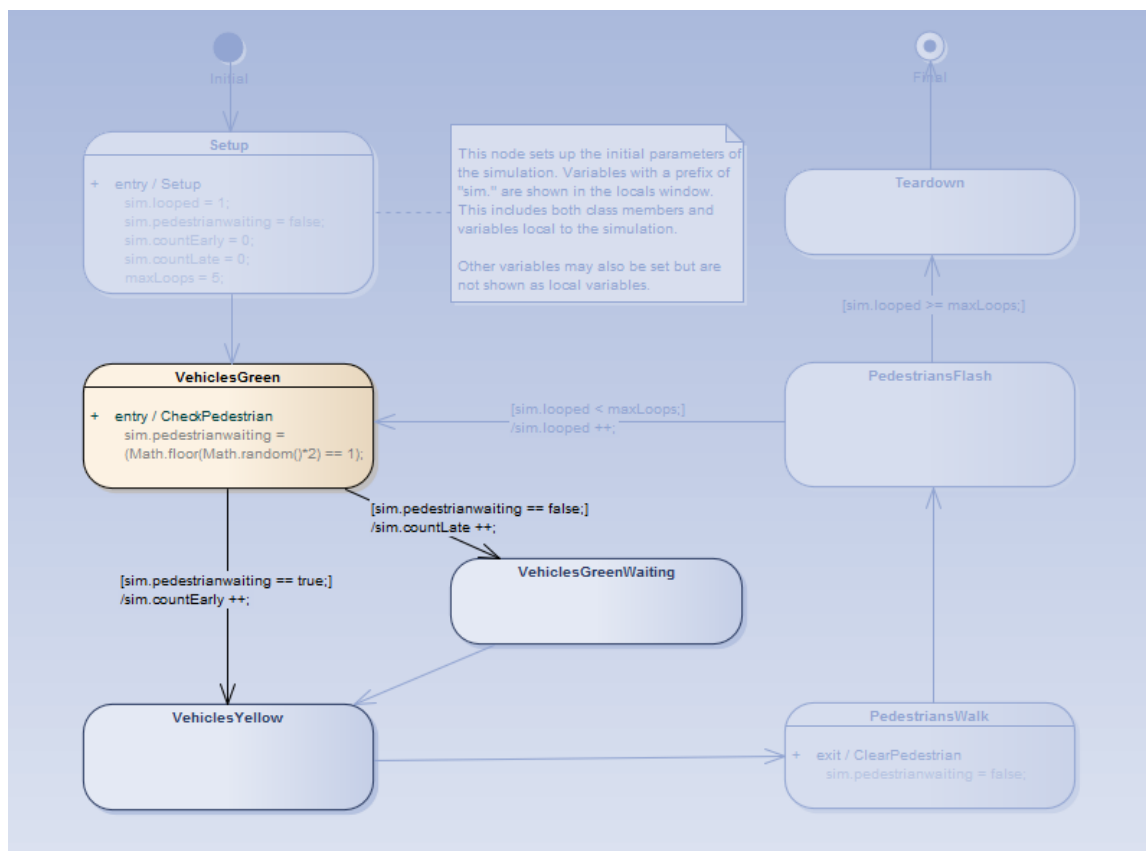
Enterprise Architect dispose d'une méthode spéciale pour afficher les informations du modèle pendant Simulation . Cela permet de concentrer l'attention sur les nœuds en cours d'exécution ou actifs.

Pendant une Simulation , Enterprise Architect suit et met en surbrillance de manière dynamique les nœuds actifs de votre modèle. Si un nœud d'un autre diagramme est activé, ce diagramme est automatiquement chargé et le nœud actuel est mis en surbrillance. Il est possible de modifier le diagramme pendant l'exécution de la Simulation . Toutefois, les modifications apportées ne sont pas reconnues tant que la Simulation en cours n'est pas terminée et qu'une nouvelle n'est pas démarrée.

Mise en évidence du ou des nœuds Actif pendant Simulation

Dans cet exemple, le nœud actuellement actif (VehiclesGreen) est mis en surbrillance dans les couleurs normales Enterprise Architect et toutes les transitions possibles hors du nœud actuel sont rendues à pleine puissance.

Les éléments qui sont des cibles possibles des transitions sortantes du nœud actif actuel sont rendus dans un style semi-estompé afin qu'ils soient lisibles et clairement différents des autres éléments du diagramme . Tous les autres éléments sont rendus dans un style entièrement estompé pour montrer qu'ils ne sont pas des cibles de l'étape Simulation suivante. Au fur et à mesure que la Simulation progresse (en particulier si elle exécute automatiquement), cette mise en évidence permet de concentrer l'attention sur l'élément actuel et son contexte visuel.



Fenêtres de Simulation

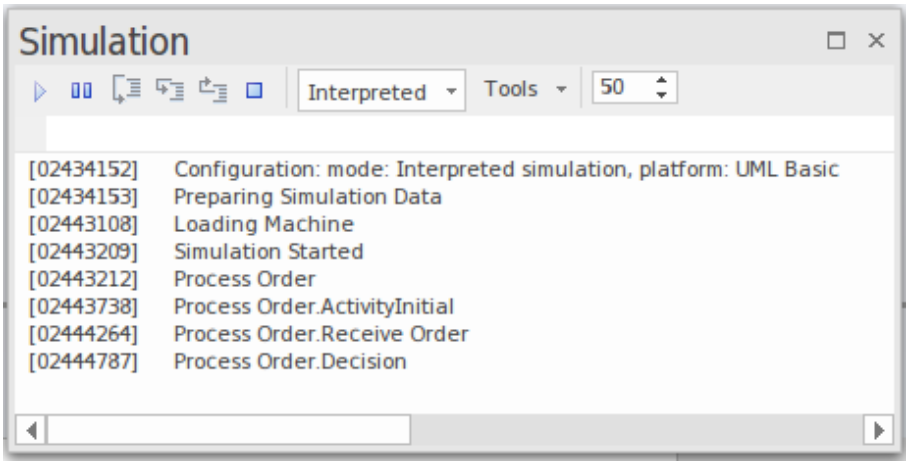
Lors de l'exécution d'une Simulation dans Enterprise Architect il est possible de définir des points d'arrêt, de déclencher déclencheurs , d'examiner des variables, d'enregistrer une trace d'exécution, de définir la vitesse Simulation , d'afficher la Pile d'Appel et de tracer visuellement les nœuds actifs au fur et à mesure du déroulement de la Simulation .

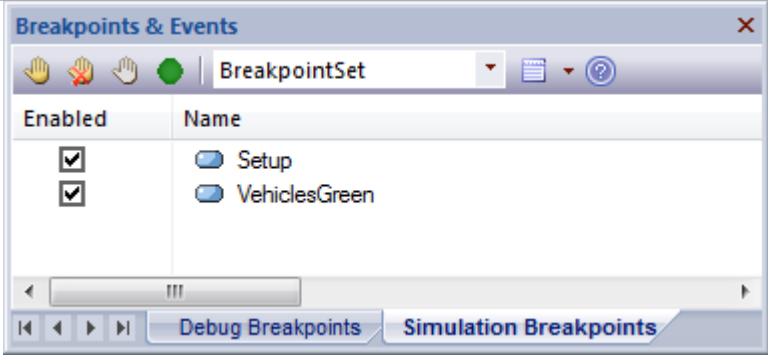
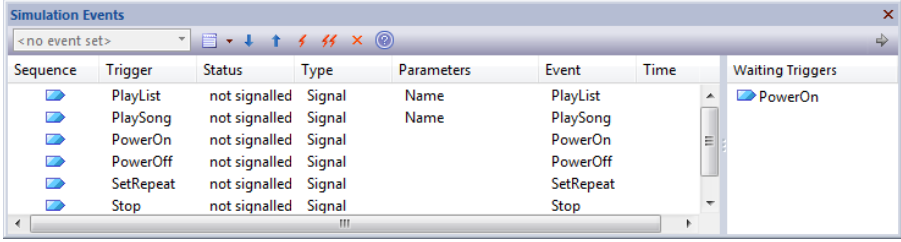
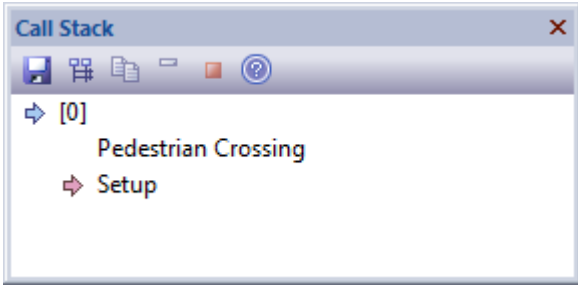
Lorsqu'une Simulation s'exécute, certains aspects tels que la sortie et l'entrée de la console se trouvent dans la fenêtre Simulation elle-même, tandis que d'autres aspects tels que les variables locales et Pile d'Appel utilisent les fenêtres standard Analyseur d'Exécution . La rubrique fournit un aperçu des principales fenêtres utilisées pendant Simulation .

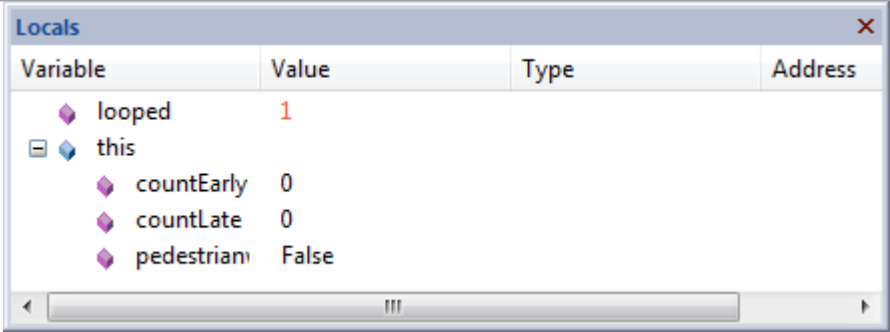
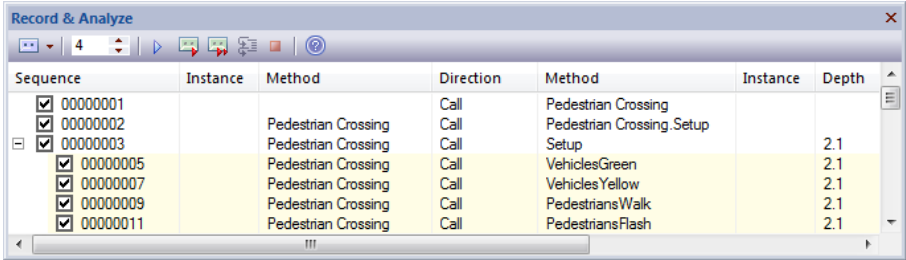
Accéder

Ruban	Simuler > Simulation Dynamique > Simulateur > Ouvrir la fenêtre Simulation
-------	--

Windows

Fenêtre	But
Exécution et console	<p>La fenêtre Simulation fournit l'interface principale pour démarrer, arrêter et parcourir votre Simulation . Pendant l'exécution, elle affiche les résultats relatifs à l'étape en cours d'exécution et d'autres informations importantes. Consultez la rubrique <i>Exécuter Simulation de Modèle</i> pour plus d'informations sur les commandes de la barre d'outils.</p> <p>Note la zone de saisie de texte juste en dessous de la barre d'outils. Il s'agit de la zone de saisie de la console. Vous pouvez y saisir des commandes JavaScript simples telles que : <code>this.count = 4;</code> pour modifier dynamiquement une variable de simulation nommée « count » à 4. De cette façon, vous pouvez influencer dynamiquement la simulation au moment de l'exécution.</p> 
Vitrine Points d'Arrêt & Événements	<p>Le processus Simulation utilise également l'onglet « Points d'Arrêt Simulation » de la fenêtre Points d'Arrêt et Marqueurs ('Simuler > Simulation Dynamique > Points d'Arrêt '). Ici, vous définissez des points d'arrêt d'exécution sur des éléments et des messages spécifiques dans une Simulation . Voir la rubrique <i>Points d'Arrêt Simulation</i> pour plus de détails.</p>

	
<p>Fenêtre Simulation Événements</p>	<p>La fenêtre Simulation Événements ('Simulate > Simulation Dynamique > Événements ') fournit des outils pour gérer et exécuter déclencheurs . Déclencheurs sont utilisés pour contrôler l'exécution des transitions Statemachine .</p> 
<p>Fenêtre de Pile d'Appel</p>	<p>Pendant la Simulation la fenêtre Pile d'Appel ('Simuler > Simulation Dynamique > Pile d'Appel ') affiche des informations sur les threads et le contexte d'exécution actuel de la Simulation .</p> <p>Le simulateur supporte les simulations multithread et inclura une entrée de thread pour chaque thread d'exécution actif et en pause. Pour chaque thread, la fenêtre Pile d'Appel affichera le contexte de départ ou d'entrée (tel qu'un élément Statemachine) ainsi que l'élément actif actuel dans ce thread. Si l'élément actif actuel est le point d'entrée d'un état composite Activity ou SubMachine, la pile inclura également l'élément actif actuel dans ce sous-contexte (et tous les autres sous-états composites actifs imbriqués).</p> 
<p>Fenêtre de variable locale Simulation</p>	<p>Le simulateur utilise la fenêtre Variables locales standard ('Simuler > Simulation Dynamique > Variables locales') pour afficher toutes les variables de simulation actuelles lorsque la simulation est en cours pas à pas ou interrompue à un point d'arrêt. Note qu'il est possible de mettre à jour dynamiquement ces variables à l'aide de la console du simulateur.</p>

	
<p>Enregistrement</p>	<p>Pendant l'exécution de votre simulation, un enregistrement de toutes les activités est conservé et affiché dans la fenêtre Enregistrer et analyser ('Exécuter > Outils > Enregistreur > Ouvrir Enregistreur '). Cela fonctionne de manière similaire à l'enregistrement normal des appels dans l'Analyseur d'Exécution Visuelle .</p> 

Configurer Script de Simulation

Vous pouvez utiliser Scripts Simulation pour contrôler précisément le démarrage d'une Simulation . En général, vous n'avez pas besoin de configurer de script Simulation sauf si :

- Vous souhaitez exécuter une Simulation interprétée qui nécessite que les variables soient initialisées avant le début de la Simulation ; cela est utile pour configurer des variables globales et définir des fonctions
- (Dans l'édition Corporate et supérieure) Vous ne souhaitez pas appliquer le comportement par défaut d'interprétation des gardes (c'est-à-dire que vous préférez utiliser une exécution manuelle), ou
- Vous souhaitez disposer de plusieurs façons d'exécuter le même diagramme

Pour la plupart diagrammes , il est possible d'initialiser un script pour une Simulation simplement en définissant des variables dans le premier élément ou connecteur après l'élément Démarrer . Pour les diagrammes State , il s'agit du connecteur Transit sortant de l'élément initial, et pour les modèles d'activité, il s'agit du premier élément Action .


Vous pouvez également utiliser Scripts Simulation pour initialiser les paramètres avant le démarrage d'une Simulation . Cela est utile pour configurer différents ensembles de valeurs initiales à l'aide de plusieurs Scripts d'Analyseur , afin de pouvoir exécuter votre Simulation dans une gamme de conditions prédéfinies.

Pour configurer un script Simulation , sélectionnez d'abord le Paquetage dans la fenêtre Navigateur , Paquetage Navigateur , Liste Diagramme ou Recherche Modèle . Vous pouvez ensuite utiliser la fenêtre Analyseur d'Exécution pour ajouter un nouveau script pour ce Paquetage sélectionné. Vous utiliserez la page ' Simulation ' de la dialogue ' Analyseur d'Exécution ' pour configurer les propriétés correspondantes.

Accéder


Affichez la fenêtre Analyseur d'Exécution en utilisant l'une des méthodes décrites ici.

Dans la fenêtre Analyseur d'Exécution , soit :

- Localisez et double-cliquez sur le script requis et sélectionnez la page « Simulation » ou
- Cliquez sur  dans la barre d'outils de la fenêtre et sélectionnez la page « Simulation »

Ruban	Develop > Source Code > Analyseur d'Exécution > Edit Scripts d'Analyseur Exécuter > Outils > Analyseur
Menu Contexte	Fenêtre Navigateur Cliquez-droit sur Paquetage Analyseur d'Exécution
Raccourcis Clavier	Maj+F12

Configurer un script Simulation

Option	Action
Plate-forme	Pour la simulation d'activité, d'interaction ou Statemachine UML , cliquez sur la flèche déroulante et sélectionnez « UML Basic ». Pour diagrammes BPMN, cliquez sur la flèche déroulante et sélectionnez « BPMN ».
Point d'entrée	Cliquez sur le bouton  et sélectionnez :

	<ul style="list-style-type: none"> • Point d'entrée pour la Simulation , et • Activité, interaction ou Statemachine à simuler <p>Si vous ne spécifiez pas de point d'entrée, le simulateur tente de parcourir l'intégralité Paquetage .</p>
Évaluer Gardes et Effets à l'aide JavaScript	<p>(Dans les éditions Corporate et supérieures) Laissez la case à cocher décochée pour effectuer une Simulation manuelle, dans laquelle vous sélectionnez l' State suivant vers lequel effectuer la transition et le point où une décision doit être prise.</p> <p>Cochez la case pour exécuter le code du comportement de l'effet dans la Simulation . La Simulation exécute le code JavaScript à ces endroits :</p> <ul style="list-style-type: none"> • Entrée/sortie/opérations d' State • Garde/effet de transition • Conditions de boucle d'activité BPMN et expressions de condition de Flux séquence <p>À l'exception de la garde, tous ces éléments doivent être une ou plusieurs instructions JavaScript valides, y compris le point-virgule.</p> <p>La garde doit être une expression booléenne valide, également terminée par un point-virgule.</p> <p>Les variables membres de « sim » ou « this » sont répertoriées dans la fenêtre Variables locales lorsqu'un point d'arrêt Simulation est atteint.</p> <pre>sim.compte = 0;</pre>
Saisir	Lorsque JavaScript est activé, vous pouvez saisir des commandes de script dans ce champ qui s'exécuteront avant l' exécuter de la Simulation .
Script de post-traitement	<p>À l'aide d'un script de post Simulation , vous pouvez exécuter JavaScript une fois la Simulation terminée. Type le nom qualifié d'un script à partir du contrôle de script du modèle.</p> <p>Par exemple, si vous avez un script nommé « MyScript » dans le groupe de scripts « MyGroup », saisissez la valeur « MyGroup.MyScript ».</p>
OK	Cliquez sur ce bouton pour enregistrer vos modifications.

Notes

- Habituellement, tous les éléments et relations Simulation résident dans le Paquetage configuré pour Simulation ; cependant, vous pouvez simuler diagrammes qui incluent des éléments de différents Paquetages , en créant des connecteurs d'importation Paquetage du Paquetage configuré vers chaque Paquetage « externe » (alternativement, pour un modèle BPSim, créez un connecteur de dépendance du Paquetage configuré vers chaque **élément** externe)

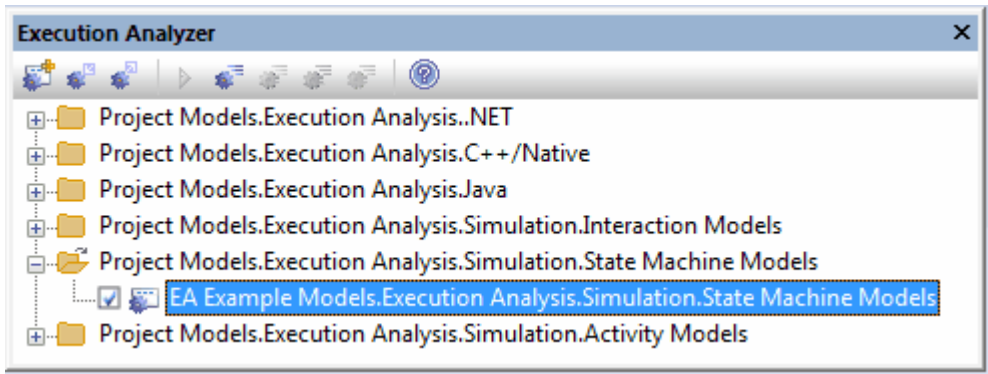
Activer Script de Simulation

Un script d'exécution est configuré pour un Paquetage de modèle définissant les paramètres Simulation . La raison la plus courante pour activer un script d'exécution est lorsque plusieurs Scripts Simulation sont configurés pour un Paquetage et que vous souhaitez exécuter un en particulier.

Accéder

Ruban	Exécuter > Outils > Analyseur Développer > Code Source > Analyseur d'Exécution
Fenêtre de l'analyseur	Cochez la case Analyzer Script pour la rendre active
Raccourcis Clavier	Maj+F12

Activer un script Simulation pour l'exécution

Étape	Action
1	<p>Dans la fenêtre Analyseur d'Exécution , sélectionnez le script d'exécution requis. Il devient alors le script par défaut de votre modèle ouvert, de sorte qu'un clic sur le bouton Simulation Exécuter invoquera automatiquement ce script Simulation .</p> 
2	Cliquez sur la case à cocher à gauche du script pour l'activer.
3	Sélectionnez l'option de ruban « Simuler > Simulateur > Ouvrir la fenêtre Simulation » pour exécuter la simulation.

Exécuter Simulation de Modèle

Une Simulation exécute le modèle étape par étape, ce qui vous permet de valider la logique de votre modèle comportemental. L'étape d'exécution en cours est automatiquement mise en surbrillance dans le diagramme du modèle pour faciliter la compréhension des différents processus et changements d'état au fur et à mesure qu'ils se produisent pendant la Simulation .

Il existe plusieurs manières de démarrer une Simulation de modèle :

- Lorsque le diagramme actif peut être simulé, le bouton Exécuter de la fenêtre principale Simulation traitera le diagramme actuel, soit en exécutant un script existant, soit en définissant un nouveau script temporaire
- Lorsque le diagramme actif ne peut pas être simulé, le bouton Exécuter de la fenêtre principale Simulation exécute la Simulation pour le script Analyseur d'Exécution actif
- En cliquant avec le bouton droit sur un script Simulation dans la fenêtre Analyseur d'Exécution et en sélectionnant l'option ' Démarrer Simulation '
- En cliquant avec le bouton droit sur un diagramme approprié et en sélectionnant l'une des options « Exécuter Simulation »

Des repères visuels sont présents pendant l'exécution. Lorsque la Simulation est en cours d'exécution, Enterprise Architect met en évidence chaque nœud actif pour chaque étape exécutée. De plus, toutes les transitions sortantes et les flux de contrôle sont mis en évidence, indiquant les chemins possibles vers l'avant. Les éléments à la fin des chemins possibles vers l'avant sont atténués de moitié et tous les autres éléments restants sont « grisés » à 90 %. Cela permet une exécution très dynamique et facile à suivre qui recentre continuellement l'attention sur le contexte d'exécution.

Accéder

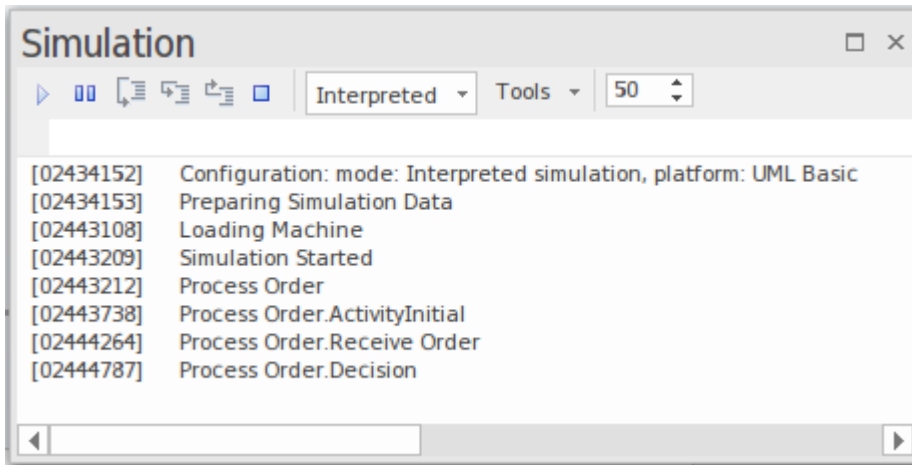
Ruban	Simuler > Simulation Dynamique > Simulateur > Ouvrir la fenêtre Simulation Simuler > Exécuter Simulation > Démarrer
-------	--





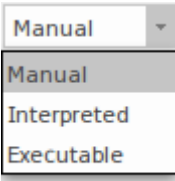
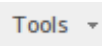

Détails spécifiques à l'édition

Dans l'édition Professional , si une branche est rencontrée lors de l'exécution, le simulateur vous propose de choisir le chemin approprié à suivre dans votre exécution.

Dans les éditions Corporate , Unified et Ultimate , dans lesquelles JavaScript est activé, la Simulation évalue automatiquement toutes les protections et tous les effets et exécute la Simulation de manière dynamique sans intervention de l'utilisateur. Si la Simulation est bloquée en raison de l'absence de chemins possibles vers l'avant évalués à True (ou de plusieurs chemins évalués à True), vous pouvez modifier les variables Simulation à la volée à l'aide de l'entrée de la console de la fenêtre d'exécution Simulation .

Exécuter une Simulation à l'aide de la barre d'outils



Icône	Action
	Démarrer le Simulateur pour le diagramme courant ou, si le diagramme courant n'est pas simulable, exécuter la Simulation en utilisant le script Simulation activé.
	Mettre la Simulation en pause.
	Lorsque la Simulation est en pause, effectuez des pas en avant, des pas en avant et des pas en arrière pour contrôler l'exécution du simulateur à l'étape requise dans la Simulation du modèle.
	Arrêtez la Simulation .
	<p>Cliquez sur la flèche déroulante et sélectionnez le type de Simulation à exécuter :</p> <ul style="list-style-type: none"> • « Interprété » - Exécuter l'exécution dynamique d'une Simulation (éditions Corporate , Unified et Ultimate) • « Manuel » : parcourez une Simulation manuellement (la seule option disponible dans l'édition Professional) • « Exécutable » - Sélectionnez lors de l'exécution de la Simulation sur un Statemachine Exécutable
	Cliquez sur la flèche déroulante et sélectionnez dans un menu d'options pour effectuer des opérations spécifiques sur le script Simulation et la sortie, telles que Build, Exécuter , Générer et Vue Points d'Arrêt .
	<p>Faire varier le taux d'exécution de la Simulation , entre 0% et 100% ; à :</p> <ul style="list-style-type: none"> • 100 % , la Simulation s'exécute à la vitesse la plus rapide possible • 0% le simulateur interrompt l'exécution à chaque instruction

Notes

- L'outil Simulation ne devient actif que lorsqu'un script d'exécution Simulation valide est activé
- Vous pouvez définir un script Simulation comme valeur par défaut actuelle en cochant sa case dans la fenêtre Analyseur d'Exécution .

Simulation Points d'Arrêt

L'onglet ' Simulation Points d'Arrêt ' de la fenêtre Points d'Arrêt & Événements vous permet d'interrompre et d'inspecter le processus Simulation .

Lors de l'exécution dynamique d'une Simulation (dans les éditions Corporate , Unified et Ultimate), le processus se déroule automatiquement. Si vous souhaitez arrêter l'exécution à un moment donné pour examiner des variables, inspecter des piles d'appels ou interagir avec le simulateur, vous pouvez définir un point d'arrêt sur un élément de modèle de la même manière que vous le feriez avec une ligne de code source. Lorsque le simulateur atteint le point d'arrêt, l'exécution est interrompue et le contrôle est renvoyé à Enterprise Architect .

Accéder

Ruban	Simuler > Simulation Dynamique > Points d'Arrêt > Simulation Points d'Arrêt
-------	---

Points d'Arrêt

La Simulation exécute le modèle étape par étape, vous permettant de valider la logique de votre modèle de comportement ; la Simulation s'arrête lorsqu'elle atteint un élément défini comme point d'arrêt.

Les éléments UML qui peuvent être définis comme points d'arrêt incluent les actions, les activités, States et la plupart des autres nœuds comportementaux tels que Décision , Initial ou Final.





Les relations UML qui peuvent être définies comme points d'arrêt incluent les messages d'interaction.

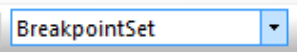

Les points d'arrêt sont stockés sous forme Ensembles Point d'Arrêt pour un projet Enterprise Architect donné.

Les éléments inclus dans une Simulation et qui ont des points d'arrêt sont signalés par un cercle vert près du coin supérieur gauche de l'élément, pendant que la Simulation est en cours. Si la Simulation n'est pas en cours d'exécution, les cercles verts ne s'affichent pas.

Lorsque JavaScript est activé, toutes les variables Simulation seront affichées dans la fenêtre Variables locales et il est possible de modifier ces variables Simulation à l'aide du champ de saisie de la console de la fenêtre Simulation (sous la barre d'outils).

Boutons de la barre d'outils

Item	Description
	Active tous les points d'arrêt définis dans l'ensemble de Point d'Arrêt actuel pour la session Simulation .
	Supprime tous les points d'arrêt définis dans l'ensemble de Point d'Arrêt actuel pour la session Simulation .
	Désactive tous les points d'arrêt défini dans l'ensemble Point d'Arrêt actuel pour la session Simulation .
	Ajoute un point d'arrêt pour l'élément sélectionné ou le message Séquence à l'ensemble Point d'Arrêt actuel.

	Modifie l'ensemble de Point d'Arrêt sélectionné à utiliser dans la session Simulation .
	Exécute les commandes Set Point d'Arrêt : <ul style="list-style-type: none">• Nouvel ensemble : Créer un nouvel ensemble Point d'Arrêt• Enregistrer sous l'ensemble : enregistre l'ensemble Point d'Arrêt actuel sous un nouveau nom• Supprimer Sélectionnée Set : Supprime le Set Point d'Arrêt actuel• Supprimer tous Ensembles : Supprime tous Ensembles Point d'Arrêt enregistrés pour le diagramme

Objets et Instances dans Simulation

Lorsqu'une activité, un système ou un processus mécanique donné s'exécute, les activités et les actions qu'il contient peuvent générer des objets d'un type spécifique et effectuer des opérations sur ces objets, voire les consommer ou les détruire. Vous pouvez simuler la création, l'utilisation et la consommation de ces objets à l'aide d'un modèle Simulation qui représente les objets et les actions avec des éléments de modèle tels que des classes, des objets d'instance, des attributs, des opérations et des ports (ActionPins et ObjectNodes). Le modèle peut également créer, agir sur et détruire plusieurs objets différents à différentes étapes dans le cadre du même processus. La représentation des données ou des objets du modèle dans Simulation permet à Simulation de refléter plus précisément le processus réel.

Concepts Object

Terme	Description
Type de Sim	Le type d'élément Simulation , tel que Classe, Énumération ou Interface. Il peut s'agir de classificateurs d'objets dans une Simulation .
SimObject	Un objet qui est une instance de (est classé par) un élément SimType.
Attribut	Une propriété d'un élément SimType ou d'un nœud spécifié tel qu'un ActivityNode.
Opération	Un comportement d'un élément SimType ou d'un nœud spécifié tel qu'un ActivityNode.
Port	Un port d'une classe ou Object , un ActionPin d'une Action ou un ObjectNode d'une activité. Les ports de classificateurs sont un type, tandis qu'un port d'un objet est une réalisation du type.
Paramètre/ Paramètre d'activité	Paramètres des opérations ; Les paramètres d'activité sont, plus précisément, les paramètres des ActivityNodes.
Fente	Une réalisation d'un attribut dans un objet . Un Slot possède une valeur de temps exécuter qui peut être initialisée par la valeur d'état exécuter du Slot. Si ces valeurs n'existent pas, le système utilise les valeurs initiales des attributs.
Environnement d'exécution	Tous les objets existent dans l'environnement d'exécution JavaScript , vous pouvez donc utiliser JavaScript pour créer ou modifier des objets de simulation et des variables de simulation.
Variables d'affichage	Tous les objets de simulation, les variables Simulation ou les événements sont identifiés dans la fenêtre Variables locales lorsqu'ils sont en vigueur. Dans certains cas, pour afficher les variables, vous devrez peut-être ajouter des points d'arrêt au modèle afin de suspendre le traitement pendant que la variable existe. Comme tous les objets et variables sont affichés, les variables globales qui existent en dehors de la simulation mais qui sont importantes pour elle (comme les éléments parent Class et Activity dans lesquels un processus est défini) sont également automatiquement représentées comme variables object par défaut. Il en va de même pour la sortie anticipée de l'activité, en tant que variable de retour.

Créer Objets dans une Simulation

Dans un modèle Simulation , vous pouvez créer des classes et soit créer des instances de celles-ci (objets globaux) pour représenter les objets qui existent dans le processus, soit définir des actions pour générer un ou plusieurs objets à tout moment pendant le processus.

Vous disposez de trois options pour créer des objets dans un modèle Simulation :

- Créer manuellement l' Object
- Créer dynamiquement un Object via un élément Action CreateObject
- Utilisez la fonction JavaScript `sim.CreateObject` (« nom ») comme « Effet » d'un élément Action , pour créer à nouveau un Object de manière dynamique

Après avoir créé un Object de manière dynamique, vous pouvez également instancier tous les objets internes de cet Object , comme une activité sur une classe, et agir sur les propriétés de cet object interne.

Créer un Object manuellement

Créez simplement un élément Object sur un diagramme dans le modèle, soit en :

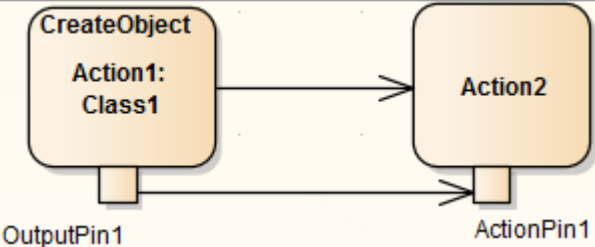
- Faire glisser un élément Object depuis les pages « Object » de la boîte à outils Diagramme et définir son classificateur, ou
- Faire glisser un élément de classificateur depuis la fenêtre Navigateur et le coller dans le diagramme en tant qu'instance

Dans le modèle Simulation , vous pouvez ensuite configurer les propriétés Object elles-mêmes (comme la définition des états d'exécution pour réinitialiser la valeur initiale d'un attribut) ou les comportements des actions pour agir sur l' Object (comme le transmettre le long d'un flux de processus) et observer ce qui arrive à l' Object dans une Simulation .

Créer un Object via une Action CreateObject

Si votre processus génère des objets lors de l'exécution, vous pouvez simuler cela à l'aide d'une Action CreateObject.

Étape	Action
1	Sur votre diagramme d'activité, faites glisser une icône « Action » depuis la boîte à outils Diagramme et sélectionnez l'option de menu contextuel « Autre CreateObject » pour la définir comme un élément Action CreateObject.
2	Définissez le classificateur de l' Action CreateObject sur la classe dont l' Object sera une instance. Cette option est définie dans la fenêtre Propriétés > CreateObjectAction > Classificateur, à l'aide du bouton [...].
3	Créez une Action Pin sur l' Action CreateObject, de type sortie.
4	Créez ou sélectionnez l' Action suivante dans la séquence de traitement et ajoutez une broche Action de type entrée. Connectez les deux actions avec un connecteur de flux de contrôle et les deux Pins Action avec un connecteur de flux Object .

	
5	<p>Effectuez une Simulation sur le diagramme . Lorsque l' Action CreateObject est exécutée, elle crée un Object ayant les propriétés du classificateur et le stocke dans sa broche de sortie. L' Object lui-même est transmis via la connexion Object Flow à la broche d'entrée de Action 2, où ses propriétés peuvent être répertoriées dans la fenêtre Locals pour la Simulation .</p>

Créer Object à l'aide de JavaScript

Vous pouvez également créer des objets Simulation de manière dynamique à l'aide d'une commande JavaScript dans le champ « Effet » de l'élément Action . La commande est :

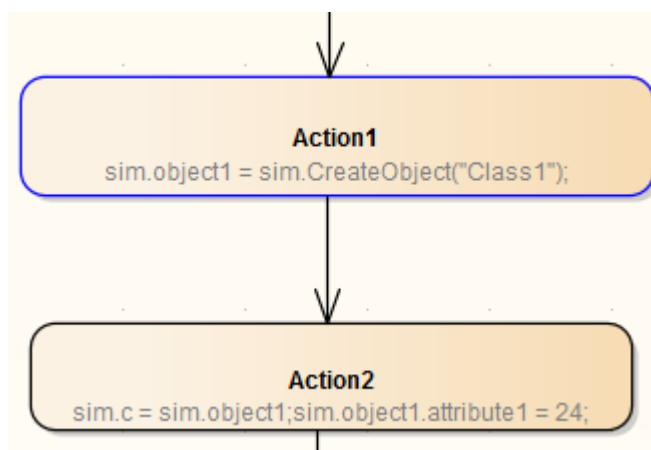
```
sim.newObject = sim.CreateObject("NomDeClasse");
```

ou

```
sim.newObject = new SimObject("ClassName"); ( JavaScript naturel)
```

C'est-à-dire : « Simuler la création d'un Object basé sur la classe <nom> ». La classe classificatrice existerait dans le même Paquetage que l' Action .

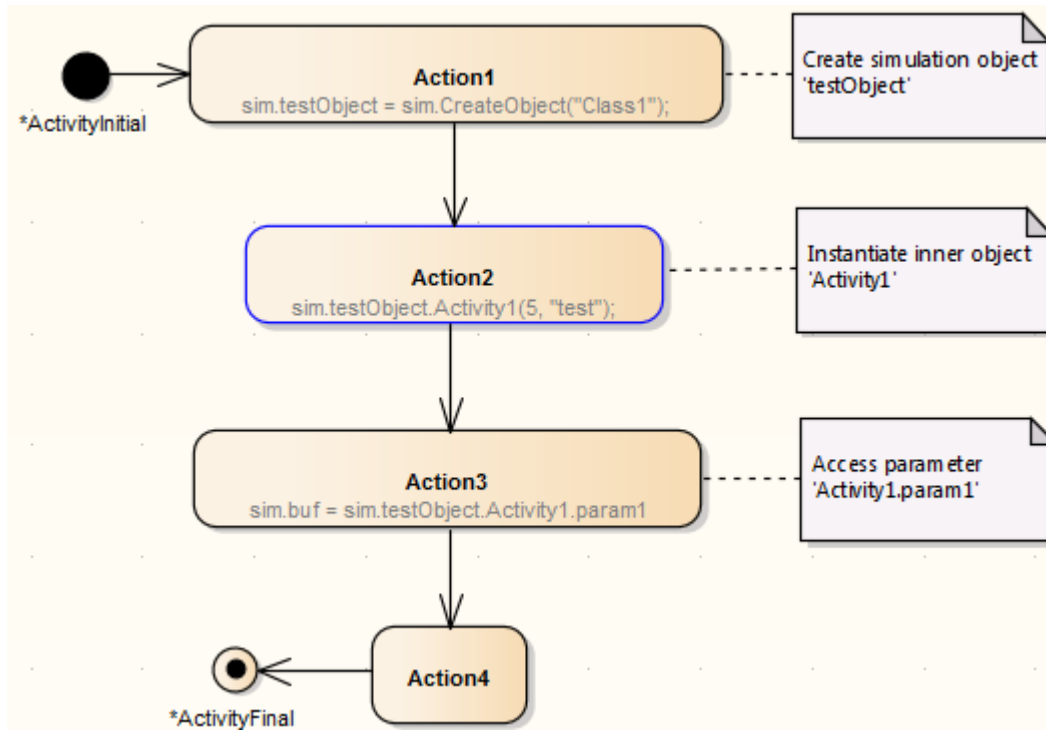
En ce qui concerne l'élément Action CreateObject, l' Object est créé pendant la Simulation et peut être transmis et traité par des éléments « en aval ». Dans cet exemple, l' Object créé est identifié comme sim.object1 et dans Action 2, il est accessible et l'un de ses attributs reçoit une valeur différente (également par JavaScript en tant qu'effet de l' Action).



Instancier des objets internes

Comme décrit précédemment, vous pouvez créer un Object à l'aide JavaScript ou d'une Action CreateObject. De même, vous pouvez instancier des objets internes à l'aide JavaScript ou d'une Action CallBehavior.

Dans cet exemple, à l'aide JavaScript , la Simulation crée d'abord un objet de test basé sur la classe 1. La classe 1 comporte un élément Activity et diagramme , avec un paramètre Activity 1 défini sur l' integer 5 et un paramètre Activity 2 défini sur la string 'test'. La valeur du paramètre Activity 1 est capturée sous la forme d'une valeur tampon 'buf'.



Détruire Objets dans une Simulation

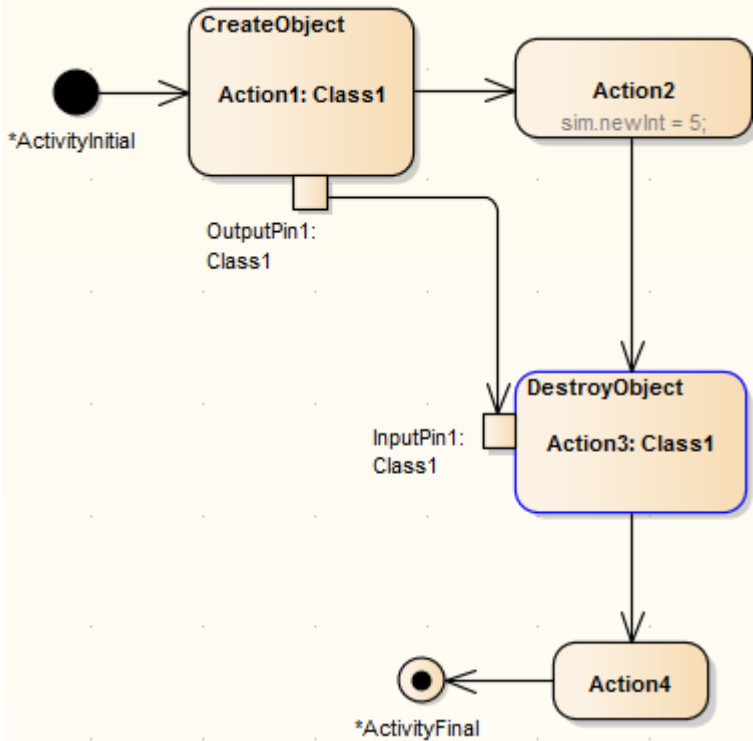
Après avoir créé ou généré des objets dans votre modèle Simulation , vous pouvez définir des actions pour détruire ces objets à tout moment du processus. Tous les objets Simulation sont détruits automatiquement une fois la Simulation terminée.

Vous avez deux options pour détruire les objets de votre modèle Simulation :

- Détruire dynamiquement les objets via un élément Action DestroyObject
- Détruire dynamiquement les objets à l'aide JavaScript dans un élément Action

Le résultat de la suppression peut être observé dans le changement des variables locales, dans la fenêtre Local.

Détruire un Object via une Action DestroyObject

Étape	Action
1	Sur votre diagramme d'activité, faites glisser une icône « Action » depuis la boîte à outils Diagramme et sélectionnez l'option de menu contextuel « Autre DestroyObject » pour la définir comme un élément Action DestroyObject.
2	Définissez le classificateur de l' Action DestroyObject sur la classe dont l' Object est une instance. (Avancé Classificateur d'ensembles). Créez une Action Pin sur l' Action DestroyObject, de type <i>input</i> .
3	Connectez la broche Action d'entrée à un connecteur de flux Object de la dernière Action ayant fonctionné sur l' Object . Dans cet exemple, la dernière Action ayant fonctionné sur l' Object est l' Action qui l'a créé. 
4	Effectuez une Simulation sur le diagramme . Le processus transmet le nom ou valeur Object à la broche Action d'entrée en tant que paramètre. Lorsque l' Action DestroyObject est exécutée, elle supprime l' Object portant ce nom ou valeur du modèle.

Dans l'exemple, l'instance de Class1 est spécifiquement détruite avant le traitement d'Action4, mais les résultats d'Action2 ne sont pas affectés.

Détruire un Object à l'aide JavaScript

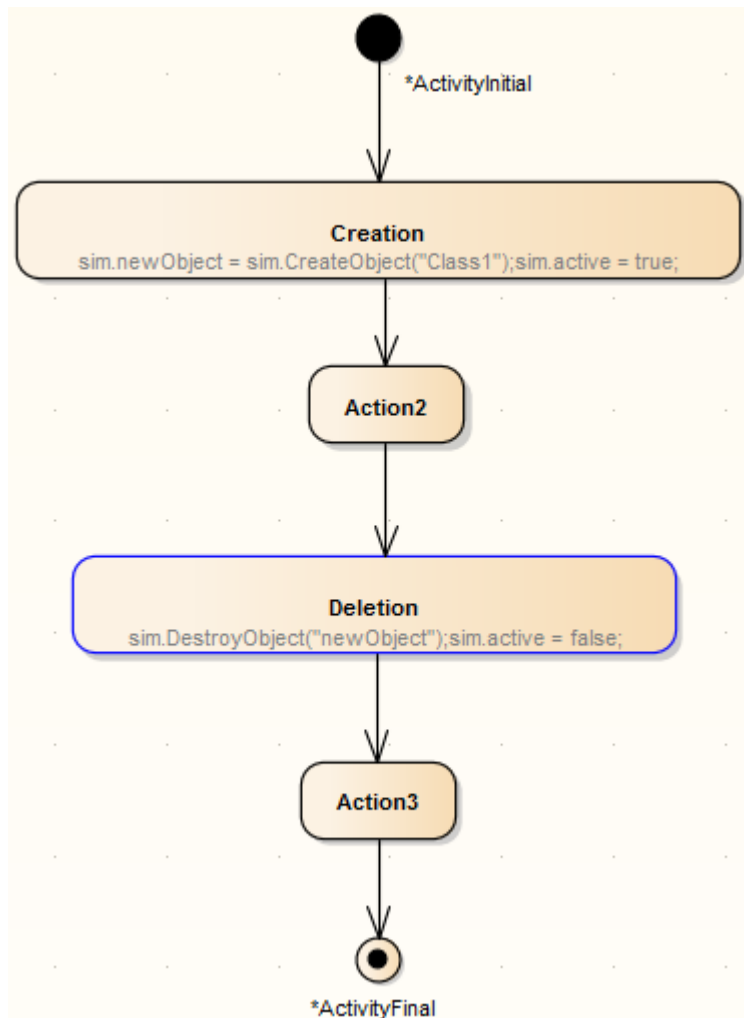
Dans la dialogue « Propriétés » de l'élément Action , dans le champ « Effet » de la page « Effet », saisissez soit :

```
sim.DestroyObject ("nom d'objet")
```

ou

```
supprimer sim.objectFullName
```

Par exemple:



Notes

- Dans les deux cas, vous pouvez également détruire un objet global (un objet créé en dehors du flux de processus) en identifiant l'Object auprès de l' Action effectuant la destruction ; dans le cas de l' Action DestroyObject, en transmettant le nom Object d'un port sur l' Object à la broche d'entrée sur l' Action via un connecteur de flux Object

Simulation dynamique avec JavaScript

Les éditions Corporate , Unified et Ultimate d' Enterprise Architect permettent d'utiliser JavaScript pour évaluer les protections, les effets et d'autres aspects du comportement dans le contexte Simulation . Cela permet une exécution entièrement automatisée et intelligente de votre modèle State ou d'activité, avec un contrôle précis des points d'arrêt, de la vitesse d'exécution et des variables Simulation .

Vous pouvez écrire JavaScript qui utilise n'importe quelle variable. Pour vous permettre d'afficher les valeurs de ces variables via l'interface utilisateur, deux objets intégrés sont définis - **sim** et **this** - dont les membres peuvent être affichés dans la fenêtre Variables locales (également appelée fenêtre Variables locales). Voici quelques exemples de variables pouvant être affichées :

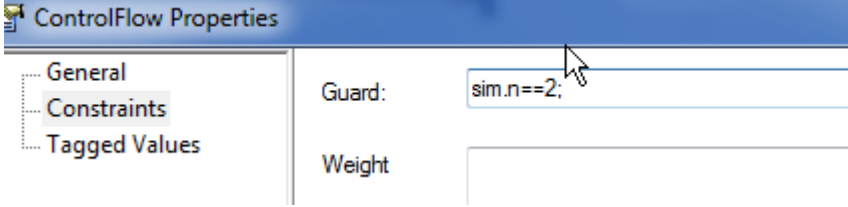
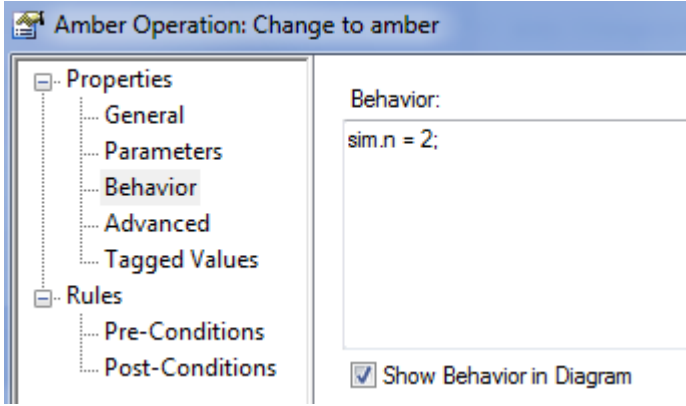
- `sim.logger`
- `sim.Nom.du.client`
- `ceci.compte`
- `ce.montant.du.compte`

La convention recommandée consiste à ajouter à l' object **sim** toutes les variables globales ou de contrôle non déclarées dans la classe propriétaire. En revanche, il serait normal d'ajouter les attributs du classificateur propriétaire à l' object **this** .

Vous trouverez ici quelques exemples de l'endroit et de la manière dont vous pouvez définir le comportement Simulation à l'aide JavaScript . D'autres exemples sont fournis dans le modèle EAExample.eap fourni avec Enterprise Architect .

Utilisation de JavaScript

Paramètre	Action
Entrée Script d'Analyseur	<p>Si vous entrez du code JavaScript dans le champ « Entrée » de la fenêtre Analyseur d'Exécution , ce code sera injecté dans la Simulation et exécuté avant le démarrage de la Simulation . Ceci est utile pour établir des variables COM, des compteurs globaux, des fonctions et d'autres initialisations.</p> <p>Platform: <input type="text" value="UML Basic"/></p> <p>Options:</p> <p><input checked="" type="checkbox"/> Evaluate Guards and Effects using JavaScript</p> <p>Input</p> <pre>sim.n=1;</pre>
Protections de flux de transition et de contrôle	<p>Il s'agit du cheval de bataille de la fonctionnalité Simulation . Comme Enterprise Architect évalue les chemins possibles vers l'avant à chaque nœud d'une Simulation , il teste les gardes sur les transitions sortantes et les flux de contrôle et n'avance que s'il existe un seul véritable chemin à suivre. Dans le cas contraire, la Simulation est considérée comme « bloquée » et une intervention manuelle est requise. Vous devez utiliser l'opérateur « == » pour tester l'égalité.</p>

	
<p>Comportement d'Éléments</p>	<p>Le comportement d'entrée et de sortie peut être défini pour States . Ce code s'exécutera au moment opportun et est utile pour mettre à jour les variables Simulation et effectuer d'autres affectations.</p>  <p>Vous pouvez également simuler le comportement des classes via leurs instances Object et leurs activités dans votre modèle.</p>
<p>Utilisation de COM</p>	<p>Une fonctionnalité très importante de l'implémentation de JavaScript dans le simulateur d' Enterprise Architect est qu'il supporte la création d'objets COM. Cela permet de connecter la Simulation en cours d'exécution à presque tous les autres processus locaux ou distants et d'influencer la Simulation en fonction de données externes ou de modifier potentiellement les données ou le comportement dans le monde extérieur en fonction de l'état actuel Simulation (par exemple, mettre à jour un modèle mécanique ou Simulation logicielle externe à Enterprise Architect). La syntaxe de création d'objets COM est présentée ici :</p> <pre> this.name="Impair"; var logger = new COMObject("MySim.Logger"); logger.Afficher(); logger.Log(" Simulation démarrée"); </pre>
<p>Utilisation Solveurs</p>	<p>Anywhere dans Enterprise Architect qui contient du code JavaScript , comme dans Simulation Dynamique , vous pouvez maintenant utiliser une construction JavaScript appelée « Solveur » (la classe Solveur) pour intégrer des outils externes et utiliser directement les fonctionnalités de chaque outil pour exécuter de manière simple et intuitive des fonctions mathématiques et graphiques complexes. Les appels vous aident à échanger facilement des variables entre le moteur JavaScript intégré et chaque environnement. Deux bibliothèques mathématiques prises en charge sont MATLAB et Octave.</p> <p>Pour utiliser la classe Solveur , vous devez avoir une connaissance des fonctions disponibles dans votre Bibliothèque mathématique préférée et des paramètres qu'elles utilisent, comme décrit dans la documentation du produit.</p> <p>Faisant partie du moteur JavaScript , ces classes Solveur sont également immédiatement accessibles aux auteurs de Add-In créant Add-Ins JavaScript basés sur des modèles.</p> <p>Consultez les rubriques d'aide <i>Octave Solveur</i> , <i>MATLAB Solveur</i> et <i>Solveurs</i> pour plus de détails.</p>

Actions signalées	<p>Il est possible de déclencher un événement signalé (déclencheur) directement en utilisant JavaScript . La commande BroadcastSignal() permet de déclencher un déclencheur nommé qui pourrait influencer l'état actuel d'une Simulation . Par exemple, ce fragment déclenche le signal (déclencheur) nommé "CancelPressed".</p> <pre>BroadcastSignal("AnnulationAppuyée");</pre> <p>Note qu'un déclencheur nommé CancelPressed doit exister dans l'environnement Simulation actuel et doit avoir ce nom de manière unique.</p> <p>Vous pouvez également appeler le signal à l'aide de son GUID. Par exemple :</p> <pre>BroadcastSignal(" {996EAF52-6843-41f7-8966-BCAA0ABEC41F} ");</pre>
EST_DANS()	<p>L'opérateur IS_IN(state) renvoie True si la Simulation actuelle a un état actif dans un thread correspondant au nom transmis. Par exemple, pour contrôler l'exécution de manière conditionnelle, il est possible d'écrire un code tel que celui-ci :</p> <pre>si (IS_IN("En attente d'entrée")) BroadcastSignal("AnnulationAppuyée")</pre> <p>Note que le nom doit être unique dans tous les contextes.</p>
Tracer()	<p>La méthode Trace(statement) vous permet d'imprimer des instructions Trace à n'importe quel moment de votre Simulation . Il s'agit d'un excellent moyen de compléter le log Simulation avec des informations de sortie supplémentaires pendant l'exécution.</p> <p>La Simulation JavaScript tronquera les chaînes qui dépassent la longueur maximale définie du message Trace.</p>

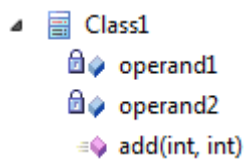
Comportements d'Appels

Au cours de la simulation d'un processus, vous pouvez exécuter les comportements définis dans une opération d'une classe (via son Object Simulation) ou d'une activité dans le modèle. Dans chaque cas, vous utilisez JavaScript pour appeler le comportement.

Invoquer le comportement d'une classe

Une classe de votre modèle définit un comportement que vous souhaitez simuler. Ce comportement est défini dans la page Comportement d'une opération de la classe.

Par exemple, la classe est destinée à additionner deux entiers, via l'opération **add(int , int)** . Les entiers dans ce cas sont des paramètres de l'opération, définis par les attributs de la classe, *operand1* et *operand2* .



Étape	Action
1	<p>Dans la fenêtre Propriétés de l'opération, sélectionnez l'onglet « Comportement » et modifiez le champ « Comportement » pour appliquer les objets Simulation JavaScript (<i>this</i> ou <i>sim</i>) à la définition du comportement.</p> <p>Dans l'exemple :</p> <pre> this.opérande1=opérande1; ceci.opérande2=opérande2; retourner opérande1+opérande2 </pre>
2	<p>Faites glisser la classe sur votre diagramme d'activité Simulation et collez-la en tant qu'instance.</p> <p>Dans l'exemple, l' Object est appelé « calculatrice ». Pour plus de clarté, l'élément affiché ici est configuré pour afficher les attributs et opérations hérités, ainsi que le code de comportement, sur le diagramme .</p> <pre> classDiagram class calculator["calculator: Class1"] { ..Class1 - operand1: int - operand2: int ..Class1 + add(int, int): int this.operand1= operand1; this.operand2 = operand2; return operand1+ operand2; } </pre>
3	<p>Sur le diagramme Simulation , pour l'élément Action approprié, ouvrez la dialogue « Propriétés » et sur la page « Effet », saisissez le JavaScript pour capturer et simuler le comportement de l' Object .</p> <p>Dans l'exemple, le JavaScript définit une valeur qui sera fournie en simulant le comportement de l'opération à partir de l' Object , telle qu'elle est effectuée sur deux entiers fournis. C'est-à-dire :</p> <pre> sim.result=sim.calculator.add(7,9) </pre>

4	Exécuter la Simulation et observez sa progression dans la fenêtre Locals. Au final, le comportement de la classe se reflète dans le résultat de la Simulation . Dans l'exemple : résultat = 16.
---	--

Invoquer le comportement d'une activité

Un élément d'activité peut avoir un comportement défini par une opération dans cet élément. À titre d'exemple simple, une activité peut avoir une opération appelée Obtenir le résultat, avec le comportement renvoyé « ON ».

Vous pouvez simuler ce comportement dans le diagramme enfant de l'activité (c'est-à-dire interne à l'activité), avec une instruction JavaScript dans le champ « Effet » de l'élément Action approprié. Dans l'exemple, cela pourrait être :

```
sim.result=ceci.GetResult();
```

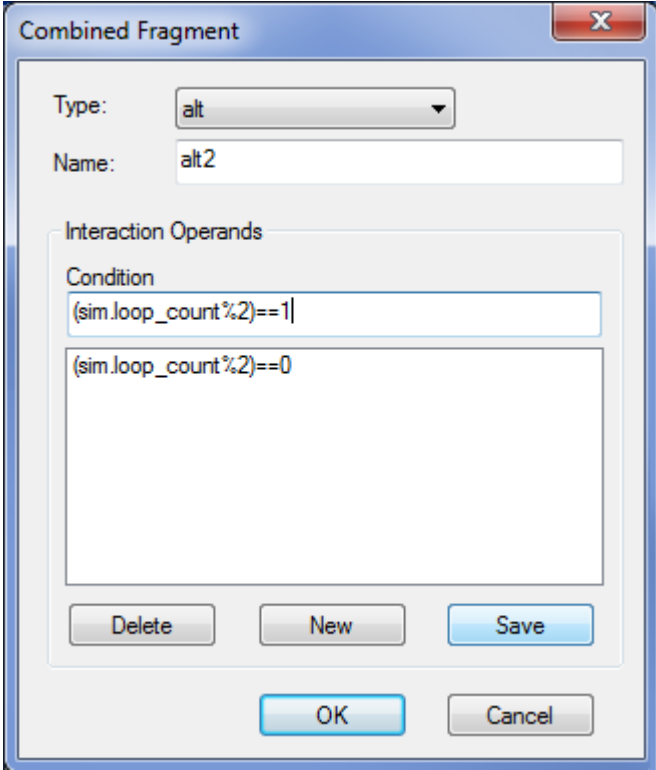
L'instruction appelle l'opération GetResult de l'activité parente et attribue le résultat du comportement de cette opération à sim.result. Vous pouvez observer la progression de la Simulation et le résultat de la simulation du comportement dans la fenêtre Variables locales, où (dans cet exemple) la valeur de résultat « ON » s'affichera finalement.

Condition d'Opérande d'Interaction et Comportement du Message

Lorsque vous simulez le comportement d'un diagramme Séquence , vous pouvez utiliser une condition pour l'opérande d'interaction CombinedFragment, afin de contrôler le flux au cours de la Simulation .

Un message dans diagramme Séquence peut être lié à une opération, de sorte que le comportement de l'opération peut également être utilisé au cours de la Simulation .

Conditions d'opérande d'interaction

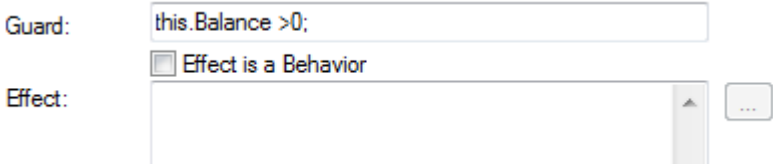
Champ/Colonne	Description
Condition d'opérande	<p>Les conditions d'opérande d'interaction sont des instructions conditionnelles qui sont évaluées chaque fois que le simulateur doit déterminer le chemin à suivre. Les conditions d'opérande ont généralement les caractéristiques suivantes :</p> <ul style="list-style-type: none"> • Défini dans la dialogue « Fragment combiné » • Écrit en JavaScript • Peut faire référence aux variables définies pendant Simulation
Ajout de conditions d'opérande	<p>Pour ajouter une condition d'opérande :</p> <ol style="list-style-type: none"> 1. Double-cliquez sur l'élément CombinedFragment pour ouvrir la dialogue « Fragment combiné ». 2. Cliquez sur le bouton Nouveau. 3. Dans le champ « Condition », saisissez le JavaScript pour la condition. 4. Cliquez sur le bouton Enregistrer. 

Sémantique d'évaluation	Pendant l'exécution, le simulateur évalue toute condition d'opérande dans le CombinedFragment ; le type CombinedFragment et le résultat de l'évaluation peuvent déterminer le chemin sur lequel la Simulation continue.
-------------------------	---

Gardes et Effets

Gardes et Effets sont utilisés pour contrôler le déroulement de la Simulation et pour exécuter des actions ou des effets supplémentaires au cours d'une Simulation .

Gardes et Effets

Concept	Détail
Gardes	<p>Les gardes sont des instructions conditionnelles qui sont évaluées chaque fois que le simulateur doit déterminer le chemin à suivre. Les gardes ont généralement les caractéristiques suivantes :</p> <ul style="list-style-type: none"> • Défini sur les transitions et les flux de contrôle pour régir le déroulement d'une Simulation • Écrit en JavaScript • Peut faire référence aux variables définies pendant Simulation
Ajout de gardes	<p>Les gardes sont définies sur la transition ou le flux de contrôle dans la dialogue « Propriétés » du connecteur sélectionné. Une garde est généralement un morceau de JavaScript qui sera évalué comme étant vrai ou faux. Par exemple, voici une instruction conditionnelle qui fait référence à une variable actuelle (Balance) supérieure à zéro. Note l'utilisation du préfixe « <i>this</i> » pour indiquer que la variable est un membre du contexte de classe actuel.</p> 
Sémantique d'évaluation	<p>Lors de l'exécution, le simulateur examinera tous les chemins possibles et évaluera les éventuelles conditions de protection. Cette évaluation pourrait établir que :</p> <ul style="list-style-type: none"> • Un seul chemin valide vers l'avant est évalué à True ; le simulateur suivra ce chemin • Il existe deux chemins valides ; le simulateur se bloquera en attendant une saisie manuelle via la fenêtre de la console pour résoudre le blocage • Aucun chemin valide n'existe ; la même réponse que lorsque deux chemins sont trouvés : le simulateur attend que l'utilisateur modifie le contexte d'exécution à l'aide de la fenêtre de la console • Aucun chemin n'est évalué à True mais une valeur par défaut (chemin non protégé) existe ; le simulateur prendra le chemin non protégé
Effets	<p>Les effets sont des comportements définis qui sont exécutés à des moments précis :</p> <ul style="list-style-type: none"> • À l'entrée dans un État • À la sortie d'un État • Lors de la transition d'un état à un autre (effet de transition) <p>Les effets peuvent être soit une section de code JavaScript , soit un appel à un autre élément de comportement dans la simulation actuelle.</p>
Effets JavaScript	<p>Un effet JavaScript pourrait ressembler à cet exemple, dans lequel la variable</p>

	<p>Balance est décrémentée :</p> <p>Guard: <input type="text"/></p> <p>Effect: <input type="checkbox"/> Effect is a Behavior this.Balance--;</p>
<p>Effets du comportement d'appel</p>	<p>Dans cet exemple, l'effet est un effet de comportement d'appel. Dans ce cas, il appelle dans un modèle l'activité nommée Decrement Balance qui est définie ailleurs. La simulation entrera alors dans ce diagramme /comportement et continuera à s'exécuter jusqu'à revenir au point auquel l'effet a été invoqué.</p> <p>Guard: <input type="text"/></p> <p>Effect: <input checked="" type="checkbox"/> Effect is a Behavior Decrement Balance </p>
<p>Ordre d'exécution des effets</p>	<p>Dans les simulations complexes qui pourraient impliquer une transition d'états profondément imbriqués vers d'autres états profondément imbriqués dans un contexte parent différent, il est important de prendre en compte ces règles concernant l'ordre d'exécution :</p> <ul style="list-style-type: none"> • Toutes les actions de sortie (effets) rencontrées en quittant un contexte imbriqué sont exécutées dans l'ordre du plus profondément imbriqué au moins profondément imbriqué • Toutes les actions (effets) définies sur les transitions sont exécutées ensuite • Enfin, tous les effets d'entrée sont exécutés du contexte le moins profondément imbriqué au plus profondément imbriqué <p>La règle de base est donc la suivante : toutes les actions de sortie, suivies de toutes les actions de transition, et enfin toutes les actions d'entrée.</p>
<p>Note sur les variables JavaScript</p>	<p>Les variables JavaScript auxquelles il faut accéder et auxquelles il faut faire référence pendant l'exécution Simulation appartiennent à :</p> <ul style="list-style-type: none"> • sim (par exemple, sim.pedestrianwaiting) - généralement utilisé pour les variables Simulation globales, ou • ceci (par exemple, this.CustomerNumber) - généralement utilisé pour faire référence aux attributs de classe propriétaires <p>Il est important de faire savoir au moteur JavaScript que vous faites référence à une variable Simulation et non à une simple variable locale utilisée, par exemple, lors de calculs de base. Vous pouvez créer des variables Simulation de portée et de profondeur arbitraires. Par exemple, this.customer.name est un nom qualifié légitime.</p>

Déclencheurs

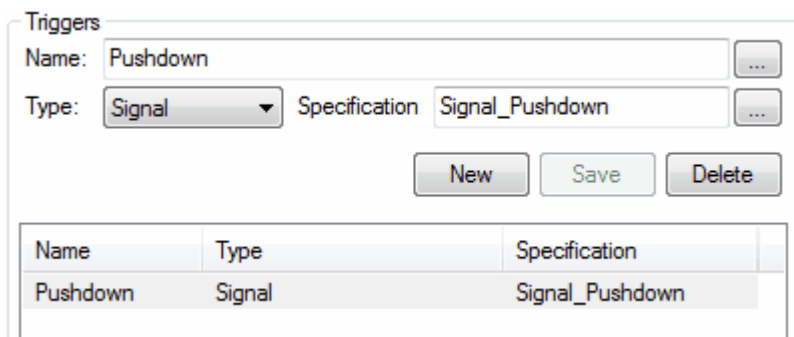
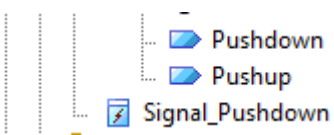
Déclencheurs représentent des signaux et des événements qui peuvent activer des transitions quittant l'état actuel. Un déclencheur peut représenter un signal ou un événement du monde réel tel que :

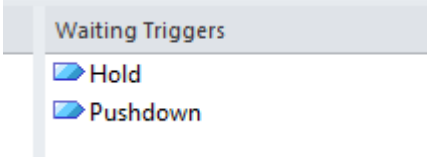
- Un bouton appuyé
- Un message en cours de réception
- Une pédale enfoncée
- Un interrupteur est actionné
- Un état dans une région concurrente en cours d'entrée ou de sortie

Pour qu'une déclencheur ait un effet

- Des transitions doivent être définies qui se déclencheront lorsque la simulation recevra le signal ou l'événement
- L' State Simulation actuel ou son ou ses parents doivent avoir une transition sortante qui accepte ce déclencheur
- La transition activée doit être non protégée ou avoir une protection qui sera évaluée à True

Gérer Déclencheurs

Action	Détail
Création Déclencheurs	<p>Déclencheurs sont créés soit comme une instance d'un élément Signal, soit comme un événement anonyme. Déclencheurs sont connectés aux transitions dans la dialogue « Propriétés de transition », comme illustré ici. Dans cet exemple, un Déclencheur nommé « Pushdown » a été défini sur la base du signal « Signal_Pushdown ».</p> <ul style="list-style-type: none"> • L'omission des détails Type et Spécification donne un simple Déclencheur anonyme. • Si des paramètres sont nécessaires, ceux-ci sont définis sur le signal et doivent être fournis au moment où l'événement se déclenche  <p>Un déclencheur apparaîtra dans l'onglet « Projet » de la fenêtre Navigateur , comme illustré ici :</p> 
Utilisation Déclencheurs	Déclencheurs sont déployés en les connectant à des transitions, comme dans

	<p>l'exemple précédent, et sont utilisés pendant la simulation en les « tirant » dans la simulation en cours d'exécution selon les besoins.</p> <p>Lors de l'utilisation déclencheurs ces points doivent être pris en compte :</p> <ul style="list-style-type: none"> • Une transition « déclenchée » ne peut pas avoir lieu tant que son déclencheur effectif n'est pas signalé ou déclenché • Lorsqu'un déclencheur est reçu, il active toutes les transitions en attente en cours dépendant de ce déclencheur (c'est-à-dire que le déclencheur est diffusé) • Déclencheurs sont évalués sur toutes les transitions pour tous les parents d'un état enfant actuel ; cela permet à un état parent de quitter tous les états enfants si nécessaire • Une fois utilisé dans une simulation, un déclencheur est consommé et doit être réactivé si nécessaire. • Ensembles de déclencheurs peuvent être enregistrés et déclenchés manuellement ou automatiquement pour faciliter la simulation automatisée de modèles sous différents modèles d'événements 															
<p>Tir Déclencheurs</p>	<p>Le déclenchement déclencheurs signifie signaler ou activer un déclencheur dans la simulation en cours. Cela peut activer zéro, une ou plusieurs transitions en attente en fonction de l'état et de la concurrence de la Simulation en cours.</p> <p>Le déclenchement déclencheurs peut être réalisé de plusieurs manières. La plus efficace est la liste « Déclencheurs en Attente ».</p> <p>Au cours de Simulation du modèle, si le simulateur atteint une impasse en raison de déclencheurs requis non disponibles (déclenchés), la liste de tous déclencheurs candidats possibles est affichée dans la liste « Déclencheurs en Attente » de la fenêtre Événements Simulation .</p>  <p>Un double-clic sur un déclencheur dans cette liste le déclenchera dans la Simulation . D'autres façons de déclencher un déclencheur incluent :</p> <ol style="list-style-type: none"> 1. Double-cliquez sur un déclencheur non signalé dans la fenêtre Événements . <table border="1" data-bbox="566 1332 1428 1500"> <thead> <tr> <th>Sequence</th> <th>Trigger</th> <th>Status</th> <th>Type</th> <th>Parameters</th> </tr> </thead> <tbody> <tr> <td>...</td> <td>Pushdown</td> <td>not signalled</td> <td>Signal</td> <td></td> </tr> <tr> <td>...</td> <td>Pushup</td> <td>not signalled</td> <td>Simple</td> <td></td> </tr> </tbody> </table> <p>Vous pouvez également utiliser le menu contextuel de ces événements pour signaler un événement non signalé ou pour signaler à nouveau un événement qui a déjà été déclenché précédemment.</p> <ol style="list-style-type: none"> 2. Utilisez le menu contextuel de la Transition requise pour tirer et sélectionnez l'option de menu 'Signal Déclencheur en Simulation '. 	Sequence	Trigger	Status	Type	Parameters	...	Pushdown	not signalled	Signal		...	Pushup	not signalled	Simple	
Sequence	Trigger	Status	Type	Parameters												
...	Pushdown	not signalled	Signal													
...	Pushup	not signalled	Simple													

Comportement d'Action par Type

Vous pouvez faire varier le comportement initié par un élément Action en définissant (ou même en redéfinissant) son type. Dans Simulation , vous pouvez appliquer et observer un certain nombre de comportements différents à l'aide des Actions des types et des groupes décrits dans ce tableau .

Types Action

Type	Description
Actions Object	<p>Les actions Object agissent sur un objet d'une manière spécifique, comme la création, la destruction ou la lecture de l' objet . Elles comprennent :</p> <ul style="list-style-type: none"> • Créer un objet • Détruire l'objet et • Lire soi-même
Actions variables	<p>Les actions variables ont une variable d'association sous la forme de la variable Valeur Étiquetée avec la valeur du nom d'un objet en cours d'exécution. Elles fournissent la variable non seulement en tant objet mais également en tant que propriété (comme un attribut ou un port) d'un objet . Elles incluent :</p> <ul style="list-style-type: none"> • Lire la variable • Écrire une variable • Variable claire • Ajouter une valeur variable • Supprimer la variable
Actions de fonctionnalité structurelle	<p>Les actions StructuralFeature agissent sur une fonctionnalité structurelle, à savoir un attribut d'une activité ou du classificateur d'un objet . Elles comprennent :</p> <ul style="list-style-type: none"> • LireStructuralFeature • Écrire une fonctionnalité structurelle • Effacer la structure • Ajouter une valeur de fonctionnalité structurelle • SupprimerStructuralFeatureValue
Actions d'invocation et d'acceptation d'événements	<p>Les actions d'invocation et d'acceptation d'événement définissent les Déclencheurs et les signaux d'un événement. Elles comprennent :</p> <ul style="list-style-type: none"> • Envoyer un signal • Signal de diffusion • Accepter l'événement • Envoyer un objet • Comportement d'appel • Opération d'appel • Accepter l'appel
Divers Actions	<p>L' Action ValueSpecification évalue une valeur ; elle doit avoir une valeur d'entrée et un code d'évaluation comme comportement ou effet.</p>

Simulation d'Activité Structurée

L'une des structures les plus complexes d'un modèle comportemental est une activité structurée, qui modélise une série d'actions soit dans une structure imbriquée, soit dans un processus d'évaluation et d'exécution. Les types d'évaluation d'une activité structurée sont le nœud conditionnel et le nœud de boucle, que vous pouvez tous deux simuler assez facilement.

Noeud conditionnel

Un nœud conditionnel se compose essentiellement d'une ou plusieurs paires de partitions Test /Corps, chaque paire étant appelée Clause. La partition Test est composée d'éléments diagramme d'activité qui testent une condition et, si cette condition est remplie, d'autres éléments de diagramme d'activité dans la partition Corps sont exécutés pour produire un résultat.

S'il existe une clause, le nœud conditionnel génère soit le résultat de la partition Body, soit aucun résultat. S'il existe plusieurs clauses, le contrôle passe d'un Test au suivant jusqu'à ce qu'une condition soit remplie et qu'une partition Body soit exécutée pour produire un résultat, ou que tous les tests échouent.

Simulation prend actuellement supporte l'utilisation de la case à cocher « Est assuré » dans l'onglet « Condition » de la fenêtre Propriétés . Les deux autres paramètres de case à cocher sont ignorés. Si la case à cocher « Est assuré » est :

- Sélectionné, au moins un Test doit être satisfait, donc son corps est exécuté et un résultat est généré ; si aucun Test n'est satisfait et aucun résultat n'est généré, le nœud conditionnel est bloqué et le traitement ne peut pas continuer au-delà
- Non sélectionné, un Test peut être satisfait et un résultat peut être généré, mais si aucun Test n'est satisfait et aucun résultat n'est généré, le traitement peut toujours se poursuivre au-delà du nœud de condition

Vous pouvez simuler une gamme de chemins et de résultats possibles en saisissant des instructions JavaScript *sim.* qui définissent ou conduisent à des résultats Test et des résultats de corps spécifiques, dans les champs « Effet » des éléments Action au sein de chaque partition de chaque clause. Ces instructions *sim.* doivent identifier le chemin complet du nœud conditionnel, de la clause et de la broche de sortie en cours de définition. Par exemple, dans un test visant à déterminer si une personne est considérée comme une personne âgée :

```
si (sim.Person.age >=65)
```

```
sim.AgeCondition.Clause1.Decider1=true;
```

```
autre
```

```
sim.AgeCondition.Clause1.Decider1=false;
```

Le nœud de condition s'appelle *AgeCondition* , le test est dans *Clause1* et le OutputPin pour ce test est *Decider1* .

Noeud de boucle

Un nœud d'activité structuré en boucle représente généralement les équivalents modélisation des instructions de boucle While, Repeat et For. Chaque nœud de boucle comporte trois partitions :

- Configuration - qui initie les variables à utiliser dans la condition de sortie de la boucle ; elle est exécutée une fois à l'entrée de la boucle
- Test - qui définit la condition de sortie de la boucle
- Corps - qui est exécuté à plusieurs reprises jusqu'à ce que le Test produise une valeur fausse

Vous définissez les nœuds de boucle en faisant glisser les éléments diagramme d'activité depuis les pages de la boîte à outils vers les partitions Configuration, Test et Corps. La partition Corps peut contenir des structures d'éléments assez complexes, définissant ce que le nœud de boucle produit réellement au cours du processus.

Le nœud de boucle possède un certain nombre de Pins Action :

- Variable de boucle (entrée) - la valeur initiale à traiter via la boucle

- Variable de boucle (sortie) - la variable variable sur laquelle le Test est effectué
- Décideur - une broche de sortie dans la partition Test , dont la valeur est examinée après chaque exécution du Test pour déterminer s'il faut exécuter le corps de la boucle
- Sortie du corps - la valeur de sortie du traitement dans la partition Corps, qui met à jour la broche de sortie de la variable de boucle pour la prochaine itération de la boucle, et
- Résultat - la valeur de l'exécution finale de la partition Test (qui est la valeur renvoyée depuis la dernière exécution de la partition Body)

Vous pouvez simuler les effets de différentes actions et sorties via la boucle, en saisissant des instructions JavaScript *sim.* qui définissent ou conduisent à des résultats Test et des résultats de corps spécifiques, dans les champs « Effet » des éléments Action au sein de chaque partition. Ces instructions *sim.* doivent identifier le chemin du nœud de boucle et de la broche de sortie en cours de définition. Par exemple, dans une Action dans la partition Test :

```
sim.LoopNode1.decider = (sim.LoopNode1.loopVariable>0);
```

Simulation de Valeur de Retour d'Activité

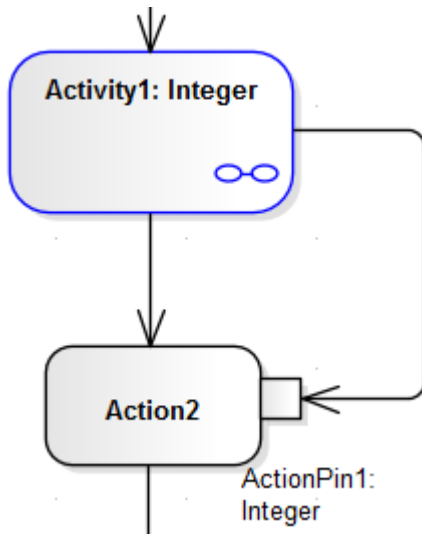
Une activité est susceptible de produire une valeur de retour en tant que sortie du processus qu'elle représente. Vous pouvez simuler la manière dont cette valeur de retour est transmise à l'étape suivante du processus, selon trois scénarios :

- L'activité produit simplement une valeur de retour, qui est transmise directement à l' Action suivante
- L'activité comporte un ou plusieurs paramètres d'activité - représentés sur un diagramme par des nœuds d'activité - qui acceptent les valeurs d'entrée ou contiennent les valeurs de sortie des actions enfants de l'activité, et le paramètre d'activité de sortie collecte et transmet la valeur de retour
- L'activité est instanciée par une Action CallBehavior qui reproduit le comportement de l'activité et transmet la valeur de retour.

Valeur de retour de l'activité Échec

(Cette méthode est unique à la simulation Enterprise Architect , imitant l'effet d'un paramètre d'activité sans qu'il n'en existe un.)

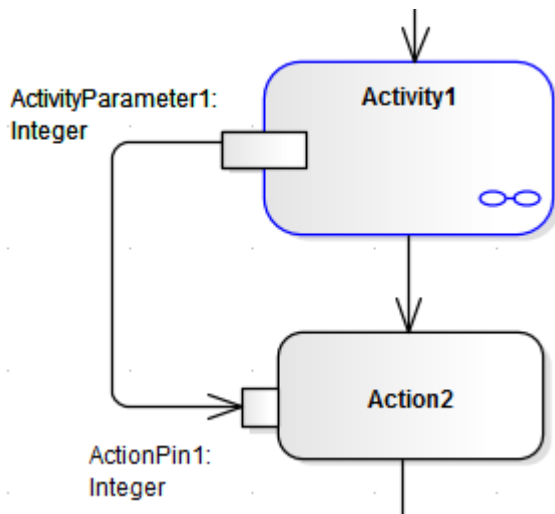
L'activité a une valeur de retour, qui est transférée de l'élément Activité à une broche Action sur l' Action suivante du processus via un connecteur de flux Object .



Vous pouvez simuler cela en définissant une instruction JavaScript simple pour définir la valeur de retour dans l'élément enfant de l'activité (comme `this.return=12;`) et, en exécutant la simulation, voir la valeur transférée à l'épingle Action dans la fenêtre Locals.

Paramètre d'activité Émission

Si l'activité possède un paramètre d'activité, sa valeur passe au nœud d'activité correspondant, puis, via un connecteur de flux Object , à l'ActionPin d'entrée de l' Action suivante du processus, comme indiqué :

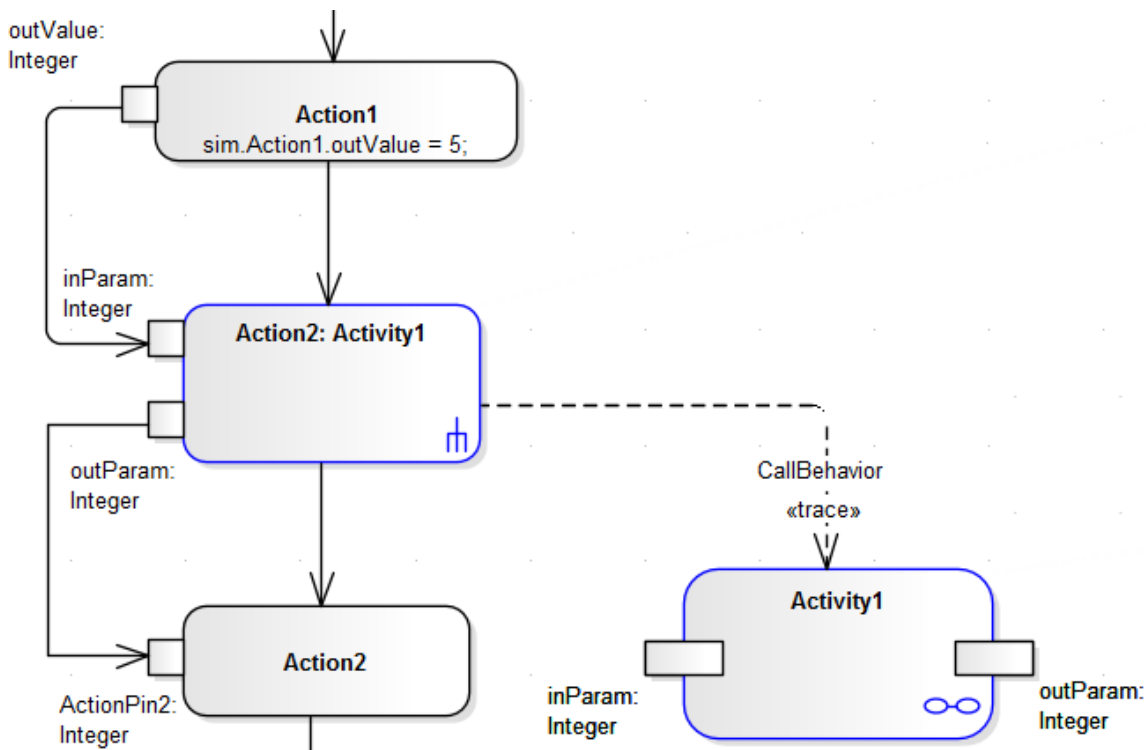


Dans la fenêtre Variables locales, vous pouvez soit observer la transmission valeur par défaut du paramètre à l'ActionPin, soit utiliser JavaScript dans les actions enfants de l'activité pour simuler une mise à jour de la valeur au sein de l'activité. Par exemple :

```
ceci.ActivityParameter1=20;
```

Action CallBehavior

Une activité peut être utilisée plusieurs fois dans un processus, auquel cas vous souhaitez peut-être générer une instance distincte de l'activité à chaque fois. Vous pouvez le faire à l'aide d'une Action CallBehavior pour créer un objet de l'activité et exécuter son comportement. Les paramètres d'activité d'entrée et de sortie sont liés aux Pins Action d'entrée et de sortie correspondantes (arguments) sur l' Action CallBehavior.



Lorsque vous simulez la partie du processus contenant l'activité, vous fournissez une valeur d'entrée (comme dans Action 1) qui passe dans la broche Action d'entrée sur l' Action CallBehavior, ce qui crée un Object de l'activité. Le comportement CallBehavior exécute le comportement de l'activité, en utilisant la broche Action d'entrée pour agir comme paramètre d'activité d'entrée et la broche Action de sortie pour recevoir le retour comme paramètre d'activité de

sortie. La valeur de retour de l'activité est ensuite transmise à une broche Action sur l' Action suivante, à l'aide d'un connecteur de flux Object . Vous pouvez fournir des instructions JavaScript dans les actions de l'activité pour agir sur la valeur d'entrée et générer une valeur de retour, telle que :

```
sim.buf=this.inParam; et
```

```
ceci.outParam=sim.buf + 11 ;
```

Fenêtre Simulation Événements

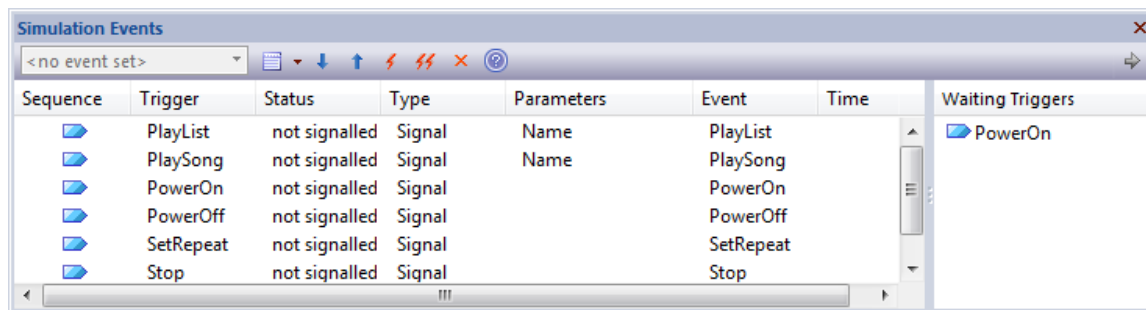
La fenêtre Événements Simulation permet de gérer déclencheurs et les ensembles d'événements d'une simulation. Ses principales fonctions sont les suivantes :

- Ajouter, supprimer et reséquencer un ensemble de déclencheurs pour une simulation
- Afficher une liste des événements déclenchés, perdus et en attente pour la simulation en cours d'exécution
- Fournir des options pour déclencher n'importe quel déclencheur arbitraire dans la simulation actuelle
- Fournir une liste pratique de « Déclencheurs en Attente » des déclencheurs que la simulation attend
- Enregistrez les ensembles déclencheur pour une utilisation ultérieure dans les simulations manuelles et automatisées
- Accepter déclencheurs glissés depuis la fenêtre Navigateur vers la liste actuelle
- Saisir les paramètres déclencheur pour un déclencheur en attente avant le déclenchement

Au fur et à mesure que déclencheurs sont consommés dans la simulation, leur statut et leur position sont enregistrés dans le corps principal de la fenêtre Événements Simulation .

Vous pouvez enregistrer le log des déclencheurs déclenchés sous forme d'ensemble déclencheur ou d'ensemble d'événements à réappliquer dans une autre Simulation exécuter , que vous pouvez exécuter manuellement ou automatiquement. Consultez la rubrique *Ensemble Déclencheur et Auto-Tir* pour plus d'informations sur la création et l'utilisation des ensembles Déclencheur .

Cette image illustre la fenêtre Simulation Événements pendant l'exécution.



Accéder



Ruban	Simuler > Simulation Dynamique > Événements
-------	---






Détails de la colonne

Champ/Colonne	Action
Séquence	Pendant et après la simulation, indique la position dans la séquence à laquelle un déclencheur a été déclenché ou est censé être déclenché. Note que si un déclencheur est déclenché hors séquence, il sera déplacé vers le bas de la section des événements signalés.
Déclencheur	Le nom du déclencheur - identifie le Déclencheur utilisé pour initier l'événement.
Statut	Indique l'état du Déclencheur . Les valeurs peuvent être :

	<ul style="list-style-type: none"> • utilisé - le déclencheur a été déclenché et le traitement a été transmis • perdu - la déclencheur a été déclenchée dans la liste, mais elle n'a eu aucun effet • signalé - un déclencheur a été déclenché et consommé par une ou plusieurs transitions • non signalé - le déclencheur n'a pas encore été actionné
Type	<p>Indique le type de déclencheur . Actuellement, seuls supporte :</p> <ul style="list-style-type: none"> • Signal • (pas de type) un déclencheur anonyme
Paramètres	<p>Pour un Signal Déclencheur , affiche initialement les paramètres requis pour le déclenchement par la spécification Signal. Par exemple, un signal « Login » peut inclure des paramètres de nom d'utilisateur et de mot de passe - et chaque invocation déclenchée peut utiliser des paramètres différents.</p> <p>A chaque fois que la simulation déclenche le déclencheur , le système vous prompt des valeurs. Vous pouvez également modifier les valeurs directement dans la liste lorsque le déclencheur est réglé sur non signalé.</p> <p>Les paramètres sont très utiles pour tester la logique conditionnelle dans votre simulation et pour simuler une variété d'entrées et de données provenant de l'extérieur de la simulation.</p>
Événement	<p>Pour un :</p> <ul style="list-style-type: none"> • Signal Déclencheur , identifie la spécification Signal • Pour Déclencheurs anonymes n'a aucune valeur
Temps	<p>L'heure de simulation à laquelle le déclencheur a été signalé. Note qu'il s'agit d'une heure absolue (du monde réel) et non d'une heure relative d'événement de simulation.</p>
Déclencheurs en Attente	<p>Répertorie les Déclencheurs disponibles pour la sélection à partir des états actuels, y compris ceux où plusieurs déclencheur sont possibles lors d'une même transition. Double-cliquez sur un déclencheur pour l'ajouter et le signaler comme déclencheur suivant dans la séquence d'événements en cours.</p> <p>Vous pouvez afficher et masquer ce panneau en cliquant sur la flèche grise juste au-dessus du panneau.</p>

Items de la barre d'outils

Option	Action
	<p>Utilisez cette liste déroulante pour sélectionner et travailler avec des ensembles déclencheur précédemment définis.</p> <p>Avant d' exécuter une simulation, sélectionnez un ensemble déclencheur préalablement défini à utiliser pour la prochaine simulation. Vous pouvez choisir de ne pas utiliser d'ensemble déclencheur en sélectionnant l'option <aucun ensemble d'événements>.</p>
	<p>Cliquez pour créer et supprimer des ensembles déclencheur :</p>

	<ul style="list-style-type: none"> • Enregistrer l'ensemble - Enregistrer la liste déclencheur actuelle en tant que nouvel ensemble déclencheur ; le système vous propose un nom pour le nouvel ensemble • Enregistrer l'ensemble sous - Créer une copie de l'ensemble actuel sous un nouveau nom d'ensemble • Supprimer Sélectionnée Set - Supprimer le set déclencheur actuel • Supprimer tous Ensembles pour Diagramme - Supprimez tous les ensembles déclencheur enregistrés pour le diagramme actuel
	<p>Déplacez le déclencheur sélectionné d'une ligne vers le bas dans la séquence de tir des déclencheurs .</p> <p>Cette option n'est pas disponible s'il n'y a pas de déclencheurs signalés sous la ligne sélectionnée.</p>
	<p>Déplacez l'entrée déclencheur sélectionnée d'une ligne vers le haut dans la séquence de déclenchement des déclencheurs .</p> <p>Cette option n'est pas disponible s'il n'y a pas de déclencheurs signalés au-dessus de la ligne sélectionnée.</p>
	<p>Cliquez pour déclencher le déclencheur sélectionné. Vous pouvez également déclencher le déclencheur en double-cliquant dessus.</p>
	<p>Cliquez pour activer et désactiver le tir automatique.</p> <p>Le déclenchement automatique déclenchera les déclencheurs non signalés de votre ensemble déclencheur de manière séquentielle. Si votre ensemble correspond à un chemin d'exécution valide, la simulation exécutera automatiquement. Les déclencheurs hors séquence ou inutilisés seront « perdus ».</p> <p>Un point d'arrêt interrompt le tir automatique et vous devrez cliquer sur le déclencheur suivant pour reprendre le tir automatique de la simulation.</p>
	<p>Supprimez le(s) déclencheur (s) sélectionné(s) de la liste.</p>

Options Menu Contexte

Option	Action
Signal sélectionné	Signalez, ou déclenchez, le déclencheur sélectionné non signalé.
Supprimer la sélection	Supprimer un déclencheur non signalé de la séquence.
Re-signaliser la sélection	Appuyer à nouveau sur un déclencheur utilisé ou signalé.
Régler tout sur Non signalé	Régler tous déclencheurs utilisés ou signalés sur non signalés.
Effacer la liste Déclencheur	Effacer tous déclencheurs de la fenêtre, quel que soit leur statut.

Déclencheurs en Attente

Lorsqu'une simulation atteint un point où tout changement d'état (pour n'importe quel thread) nécessite qu'un Déclencheur se déclenche, la simulation est effectivement mise en pause et le contrôle revient au système. La simulation attend maintenant effectivement une forme d'événement (un signal du monde réel) pour se déclencher. La liste Déclencheurs en Attente est utile pour aider à déterminer quel Déclencheur doit être signalé manuellement.

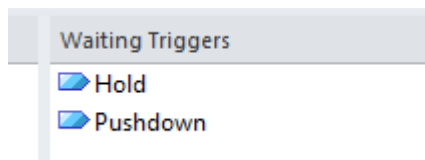
Accéder

Ruban	Simuler > Simulation Dynamique > Événements Le volet de droite répertorie Déclencheurs disponibles.
-------	--

La liste Déclencheurs en Attente sur la fenêtre Simulation Événements est la suivante :

- Rempli à chaque cycle Simulation avec tous Déclencheurs qui auraient un effet immédiat s'ils étaient signalés
- Rempli d'un ensemble discret (les doublons éventuels ne sont pas affichés car un Déclencheur est effectivement diffusé à toutes les transitions à la fois)
- Activé en double-cliquant sur le Déclencheur d'intérêt
- Inclut tous déclencheurs possibles - y compris ceux qui activent les transitions sur les parents des états actuellement imbriqués

Cet exemple montre que la simulation actuelle a atteint un point où deux Déclencheurs possibles peuvent influencer le flux d'exécution.



En raison de la nature des Déclencheurs et de leurs effets, la liste peut se référer à chacune de ces situations d'exemple de manière tout aussi valable :

- Un seul état a deux transitions sortantes qui attendent respectivement Hold et Pushdown ; le déclenchement de l'une d'entre elles activera la transition associée dans la simulation
- Un même état a deux ou plusieurs déclencheurs possibles pour la même transition, comme une caméra de sécurité allumée par un détecteur de mouvement, un détecteur de son ou un détecteur de chaleur.
- Deux (ou plusieurs) threads (régions simultanées) ont chacun un état en attente de Hold ou Pushdown ; le déclenchement de l'un de ces déclencheurs entraînera l'attente du ou des threads sur ce déclencheur pour continuer tandis que les autres threads resteront bloqués
- Un état enfant attend l'un des déclencheurs tandis qu'un état parent attend l'autre ; le déclenchement d'un déclencheur entraînera le déclenchement de la transition associée et l'enfant ou le parent procédera en conséquence.
- Toute combinaison de ceux-ci

Déclencheurs de Re-Signal

Il est possible de re-signaliser un Déclencheur comme raccourci pour faire glisser des instances Déclencheur supplémentaires pour la signalisation.

Accéder

Affichez la fenêtre Simulation Événements , puis cliquez-droit sur un Déclencheur dans cette fenêtre et sélectionnez l'option 'Re-Signal Selected'.

Ruban	Simuler > Simulation Dynamique > Événements > cliquez-droit sur déclencheur existant > Re-Signal sélectionné
-------	--

Liste Déclencheur

La fenêtre Simulation Événements contient une liste de Déclencheurs qui ont déjà été déclenchés. En cliquant avec le bouton droit de la souris sur un Déclencheur que vous souhaitez signaler à nouveau, vous pouvez utiliser le menu contextuel pour provoquer le déclenchement à nouveau.

Cette image montre la re-signalisation en action. Lorsqu'un signal est re-signalé, une nouvelle copie est créée et placée à la fin de la liste déclencheurs signalés, où elle est automatiquement déclenchée à nouveau.

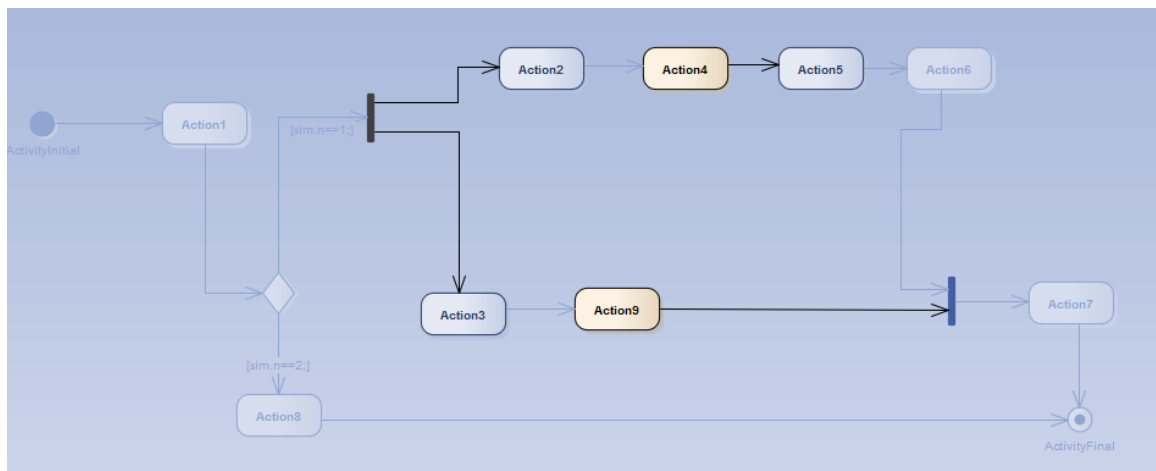
Sequence	Trigger	Status	Type	Parameters
1	Pushdown	used	Signal	
2	Pushup	used		
3	Pushdown	used		
4	Pushup	used		

- Signal Selected
- Removed Selected
- Re-Signal Selected**
-
- Set All to Unsignalled
- Clear Trigger List

Multi-filetage - Fourches et Jointures

Le simulateur Modèle offre la possibilité de gérer des simulations multithreads à l'aide de nœuds Fourche et Joindre .

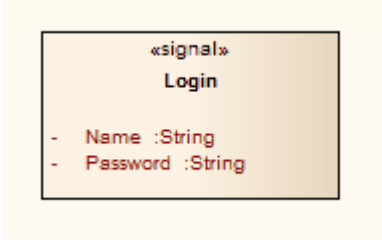
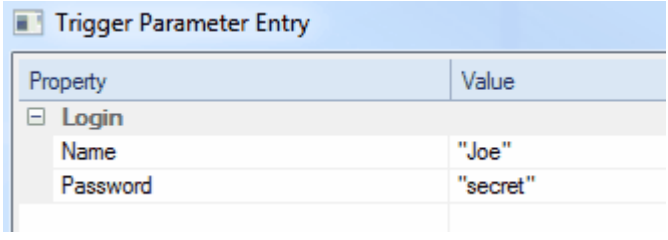
- Dans l'exemple, le point d'exécution actuel s'est divisé en deux threads, chacun avec son propre nœud actif
- Au fur et à mesure que cet exemple progresse, la branche inférieure attendra au niveau du nœud Join jusqu'à ce que la branche supérieure ait terminé toutes ses actions
- Une fois que les deux threads fusionnent à nouveau en un seul, la Simulation se poursuivra comme un seul thread jusqu'à la fin.
- Lors de l'exécution automatique, chaque thread sera considéré comme exécutant une seule étape au cours d'un « cycle » de simulation - bien que lors d'une exécution pas à pas unique ou à un point d'arrêt, le comportement consiste à alterner l'exécution pas à pas entre les threads lorsque chaque thread reçoit du temps de traitement
- Note que la fenêtre Pile d'Appel affichera deux threads actifs et un thread « en pause » dans l'exemple ; une fois les threads fusionnés, il y aura un retour à l'exécution à thread unique
- note également que les variables locales sont partagées (globales) entre tous les threads ; si vous souhaitez simuler des variables privées sur un thread, vous devez créer de nouvelles variables Simulation au début de chaque thread, en préchargeant ces variables avec les données globales existantes.

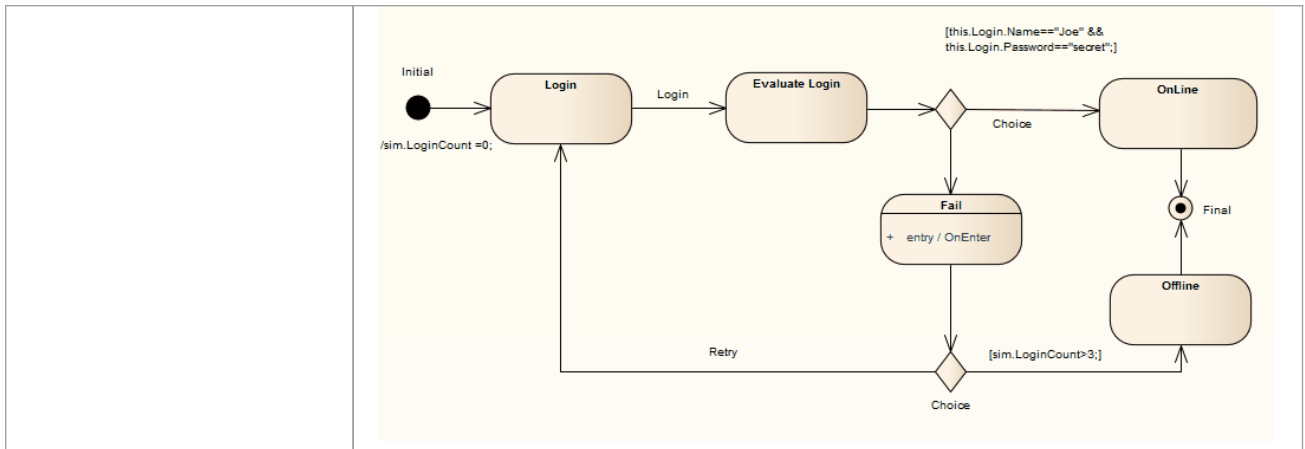


Paramètres Déclencheur

Les paramètres Déclencheur sont des arguments passés dans la simulation avec un déclencheur lorsqu'il est déclenché. Ils permettent de spécifier un comportement complexe et de prendre des décisions en fonction des variables et des données passées dans une simulation au moment de l'exécution par un déclencheur déclenché (événement).

Paramètres

Paramètre	Détail
Introduction	<p>Pour utiliser les paramètres déclencheur vous devez :</p> <ul style="list-style-type: none"> • Créez d'abord un élément Signal avec les attributs appropriés qui deviendront vos paramètres au moment de l' exécuter • Sur une transition appropriée dans votre diagramme , créez un déclencheur basé sur le signal créé précédemment • Au moment de l'exécution, il sera demandé de saisir les paramètres appropriés - ils sont ensuite transmis avec le déclencheur
Signaux	<p>Un élément Signal est un gabarit ou une spécification à partir duquel déclencheurs réels peuvent être construits. Cet exemple comporte deux arguments, un nom et un mot de passe. Ceux-ci seront renseignés au moment de l'exécution, soit manuellement, soit dans le cadre d'un ensemble déclencheur prédéfinis.</p>  <pre> classDiagram class Signal { Name :String Password :String } </pre>
Paramètres Déclencheur	<p>L' prompt des paramètres Déclencheur demande des valeurs appropriées pour chaque paramètre. Note que vous devez placer les chaînes entre guillemets, sinon l'interpréteur pensera que vous faites référence à d'autres variables.</p> 
Exemple Diagramme	<p>Il s'agit d'un exemple diagramme qui utilise des paramètres déclencheur . À l'état Évaluer la connexion, la simulation examine les variables transmises en tant que paramètres déclencheur et prend la décision d'accepter les informations d'identification ou de les refuser.</p>



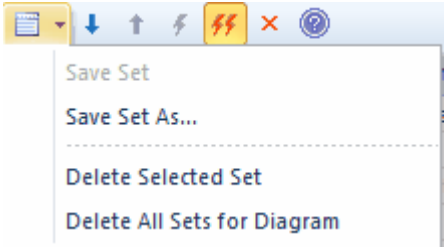
Ensemble Déclencheur et Auto-Tir

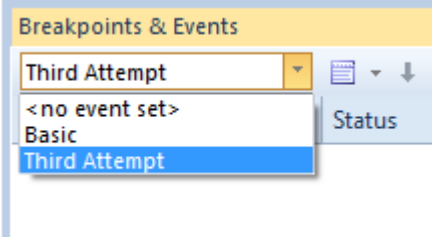
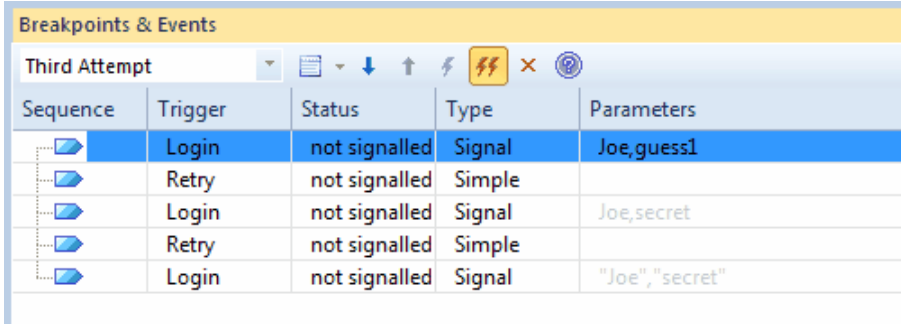

Ensembles Déclencheur sont un moyen efficace d'automatiser et de rationaliser l'exécution, les tests et la validation des modèles de simulation. En réutilisant des ensembles de déclencheurs (avec ou sans paramètres), il est possible de parcourir rapidement et efficacement de nombreux scénarios de simulation, soit manuellement, soit automatiquement à l'aide de l'outil « auto-firing ».

Accéder

Ruban	Simuler > Simulation Dynamique > Événements
-------	---

À propos des Ensembles de Déclencheurs

Aspect	Détails
Ensembles Déclencheur	<ul style="list-style-type: none"> • Stocké avec un diagramme associé • Constitué d'une liste de Déclencheurs dans une séquence définie • Peut inclure des paramètres Déclencheur si nécessaire • Peut être utilisé manuellement en double-cliquant sur Déclencheurs pour tirer selon les besoins • Peut être utilisé dans le cadre du comportement « tir automatique » pour automatiser l'exécution • Géré depuis la fenêtre Simulation Événements
Gestion Ensembles	<p>Les ensembles Déclencheur peuvent être créés en faisant glisser manuellement déclencheurs dans la liste déclencheurs actifs, puis en utilisant le menu déroulant « Gérer Ensembles Déclencheur » pour enregistrer un nouvel ensemble.</p> <p>Il est également possible de sauvegarder un ensemble de déclencheurs créés au cours d'une même simulation en tant que nouvel ensemble. Cela est pratique pour créer plusieurs chemins de test au cours d'une simulation, en sauvegardant les déclencheurs déclenchés manuellement pour chaque cas de test.</p>  <p>Vous pouvez également supprimer un ensemble et supprimer tous les ensembles du diagramme actuel.</p> <p>Il est également possible de charger un ensemble, de modifier les paramètres et/ou l'ordre de déclenchement et de sauvegarder l'ensemble sous un nouveau nom. Il s'agit d'une méthode pratique pour créer rapidement une suite de scripts de test de simulation.</p>
Utilisation Ensembles	Pour utiliser un ensemble déclencheur sélectionnez-le d'abord par son nom dans la

	<p>liste déroulante des ensembles déclencheur , comme dans cette image d'exemple. Une fois sélectionné, il charge la fenêtre Liste Déclencheur avec l'ensemble déclencheur défini.</p> <p>Note que l'élément spécial <i><no event set></i> signifie qu'aucun ensemble n'est actuellement sélectionné. Au début de chaque simulation, si un ensemble est sélectionné, il sera chargé à nouveau pour le prochain exécuter . Si <i><no event set></i> est sélectionné, la liste déclencheur sera effacée.</p>  <p>Une fois que vous avez sélectionné un ensemble déclencheur et que la liste des déclencheurs est chargée, vous avez deux options :</p> <ul style="list-style-type: none"> • Actionnez les déclencheurs manuellement selon les besoins • Utilisez la fonctionnalité de tir automatique pour automatiser entièrement la simulation  <table border="1" data-bbox="517 860 1422 1182"> <thead> <tr> <th>Sequence</th> <th>Trigger</th> <th>Status</th> <th>Type</th> <th>Parameters</th> </tr> </thead> <tbody> <tr> <td>...</td> <td>Login</td> <td>not signalled</td> <td>Signal</td> <td>Joe,guess1</td> </tr> <tr> <td>...</td> <td>Retry</td> <td>not signalled</td> <td>Simple</td> <td></td> </tr> <tr> <td>...</td> <td>Login</td> <td>not signalled</td> <td>Signal</td> <td>Joe,secret</td> </tr> <tr> <td>...</td> <td>Retry</td> <td>not signalled</td> <td>Simple</td> <td></td> </tr> <tr> <td>...</td> <td>Login</td> <td>not signalled</td> <td>Signal</td> <td>"Joe","secret"</td> </tr> </tbody> </table>	Sequence	Trigger	Status	Type	Parameters	...	Login	not signalled	Signal	Joe,guess1	...	Retry	not signalled	Simple		...	Login	not signalled	Signal	Joe,secret	...	Retry	not signalled	Simple		...	Login	not signalled	Signal	"Joe","secret"
Sequence	Trigger	Status	Type	Parameters																											
...	Login	not signalled	Signal	Joe,guess1																											
...	Retry	not signalled	Simple																												
...	Login	not signalled	Signal	Joe,secret																											
...	Retry	not signalled	Simple																												
...	Login	not signalled	Signal	"Joe","secret"																											
<p>Tir automatique</p>	<p>Le déclenchement automatique est un moyen pratique de rationaliser vos simulations. Une fois que vous avez chargé un ensemble déclencheur , si vous sélectionnez le bouton de déclenchement automatique  , Enterprise Architect sélectionnera automatiquement déclencheurs en attente lorsqu'il atteindra une impasse dans la simulation. En pratique, cela signifie que les ensembles déclencheur correspondant exactement à un chemin dans la simulation exécuter automatiquement sans votre intervention.</p> <p>Comme vous pouvez enregistrer n'importe quel nombre d'ensembles de déclencheur avec différents chemins et paramètres déclencheur , vous pouvez tester et travailler efficacement et rapidement avec de nombreux scénarios différents.</p>																														
<p>Règles de tir automatique</p>	<p>Lorsqu'une simulation s'exécute avec le déclenchement automatique activé, Enterprise Architect attend jusqu'à ce qu'un point soit atteint où la simulation est « bloquée » ou stable, en attendant qu'un ou plusieurs déclencheurs fassent avancer la simulation. À ce moment-là, le premier déclencheur non déclenché de la liste sera sélectionné et déclenché dans la simulation. Le résultat dépend de la pertinence et peut-être des paramètres du déclencheur .</p> <ul style="list-style-type: none"> • Si le déclencheur correspond à un déclencheur « en attente », il est immédiatement consommé et la simulation avance • Si le déclencheur ne correspond à aucun déclencheur « en attente » ou à une transition parent possible, alors le déclencheur est « perdu » et la simulation reste dans l'état actuel ; cela correspond à un scénario tel qu'un utilisateur appuyant plusieurs fois de suite sur un bouton « marche » - il n'y a aucun effet autre que celui occasionné par la première pression 																														

Utiliser Ensembles Déclencheur pour simuler une Séquence d'événements

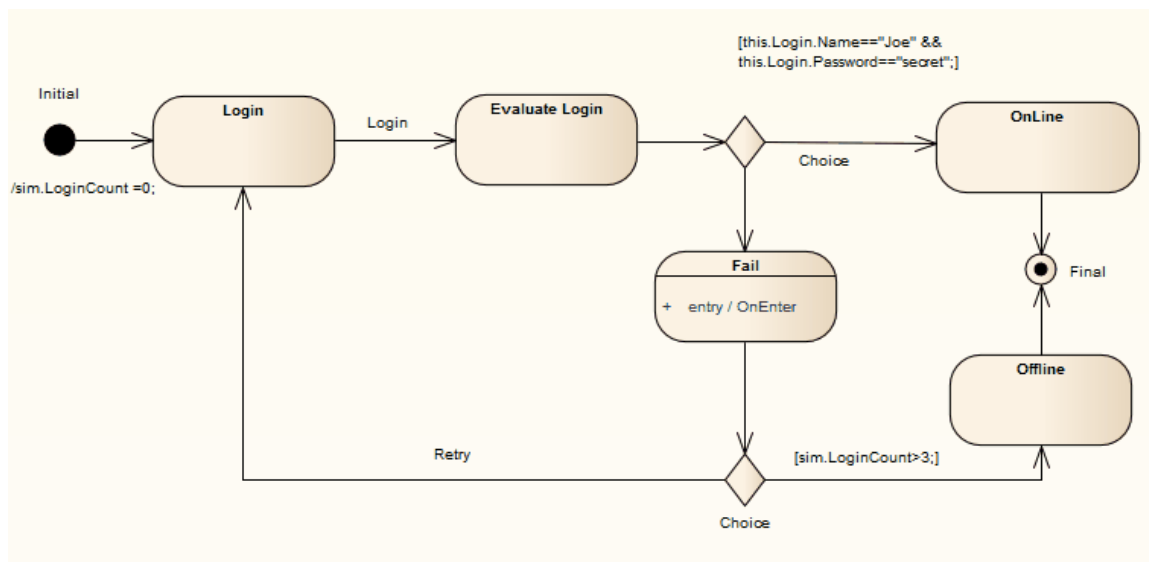
À titre d'exemple simple de l'utilité des ensembles déclencheur, considérons cet exemple d'ensemble déclencheur et diagramme associé.

Dans cet exemple, en utilisant un nom d'utilisateur et un mot de passe, nous simulons un processus de connexion simple de type « trois coups et vous êtes dehors ». Le chemin de réussite attend le nom « Joe » et le mot de passe « secret ». (Note : il est très important que les paramètres référençant des chaînes soient placés entre guillemets, sinon l'interpréteur pense que le nom fait référence à une autre variable dans la simulation.)

- Pass 1 essaie *Joe* et *guess1* - qui échoue
- Le passage 2 essaie *Joe* et *secret*, mais comme ils font référence à des variables et non à des chaînes, cela échoue également
- Le passage 3 montre la manière correcte de référencer les paramètres déclencheur et la simulation réussira

Breakpoints & Events				
Third Attempt				
Sequence	Trigger	Status	Type	Parameters
1	Login	not signalled	Signal	Joe,guess1
2	Retry	not signalled	Simple	
3	Login	not signalled	Signal	Joe,secret
4	Retry	not signalled	Simple	
5	Login	not signalled	Signal	"Joe", "secret"

Voici un diagramme simple simulant un processus de connexion nécessitant une paire nom d'utilisateur et mot de passe.



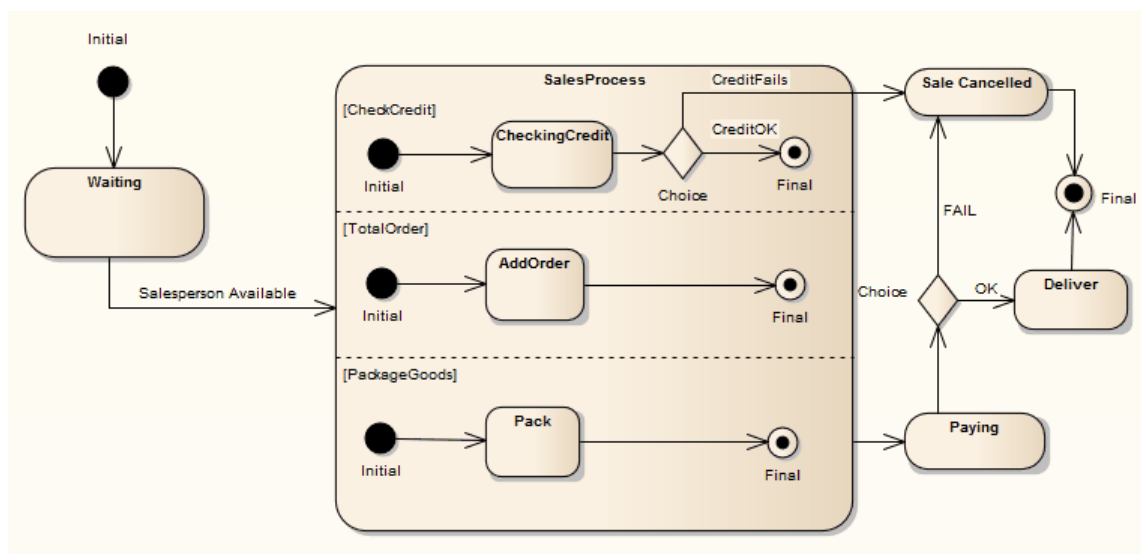
Multi-filetage - Régions State concurrentes

Les régions concurrentes au sein d'un State représentent les changements d'état et le traitement qui se produisent en parallèle au sein d'un State parent global. Cela est particulièrement utile lorsqu'une région déclenche des événements ou modifie des variables de simulation dont dépend une autre région. Par exemple, une région peut contenir un minuteur simulé qui déclenche des événements à des intervalles définis qui invoquent des changements d'état dans les States d'autres régions.

Les régions simultanées sont essentiellement les mêmes que les fourches et Jointures avec une logique et des règles de traitement similaires.

Dans l'exemple :

- Lorsque la transition vers SalesProcess est effectuée, chaque région est activée simultanément
- Le crédit est vérifié, la commande totalisée et les marchandises requises emballées
- Cependant, en cas d'échec de la vérification de crédit, cela déclencheurs la transition vers l'état Vente annulée ; note que lorsque cela se produit, l'ensemble de l'état parent et toutes les régions détenues sont immédiatement quittés, quel que soit leur état de traitement
- Si la vérification de crédit réussit, la région passe à l'état final et une fois que les autres régions ont toutes atteint leur propre état final, l'état parent peut alors être quitté

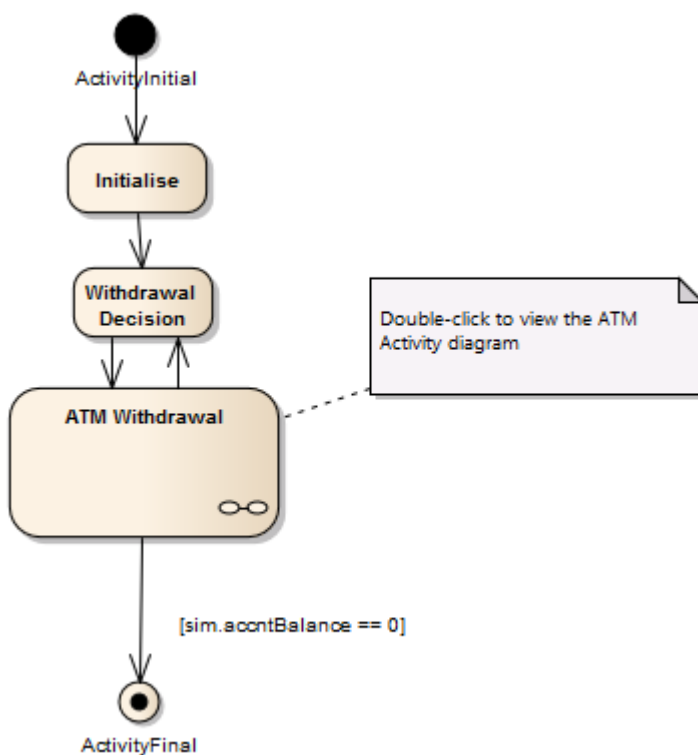


Utilisant Diagrammes composites

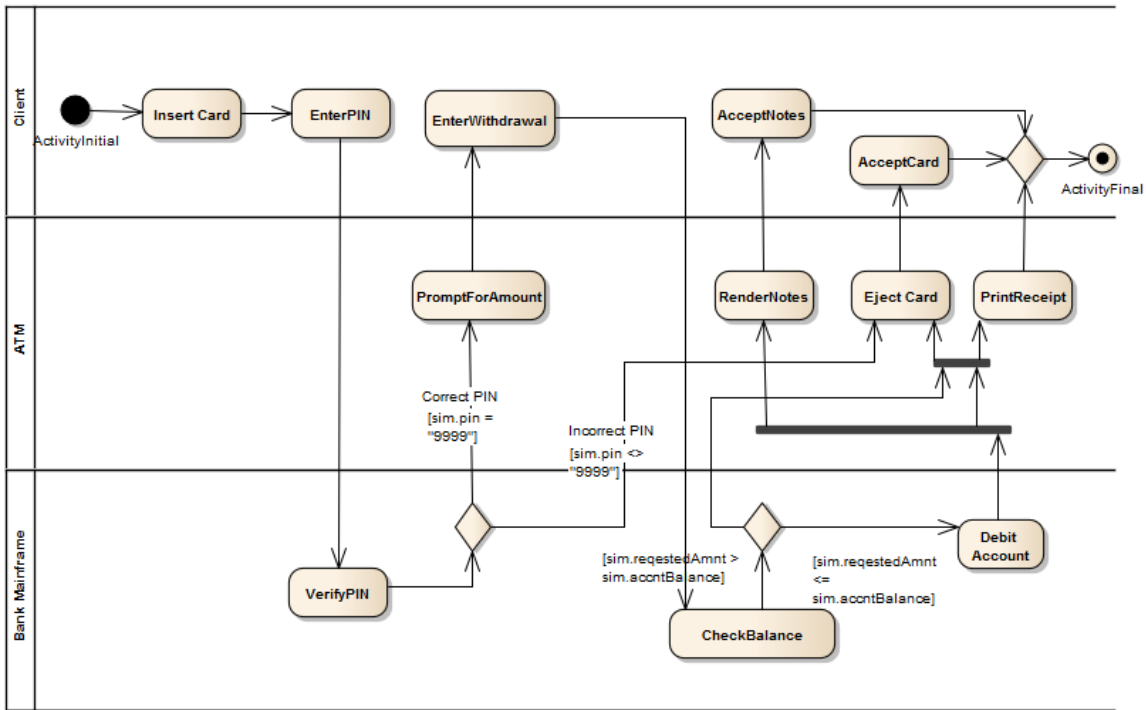
Si vous souhaitez simuler un traitement qui inclut une branche représentée sur un diagramme différent (par exemple, pour réduire la complexité sur le diagramme principal ou pour masquer les zones de traitement qui ne sont traitées qu'en cas d'exception), vous pouvez utiliser un élément Composite pour représenter et accéder à la branche sur son diagramme Composite enfant. Lorsque vous exécutez la simulation et qu'elle atteint l'élément Composite, elle ouvre le diagramme enfant et le traite avant de revenir (le cas échéant) au chemin de traitement principal. Il s'agit d'une excellente méthode pour suivre le chemin de traitement dans un processus complexe, en représentant des sections du processus avec des éléments d'activité composite qui développent le traitement réel dans leurs diagrammes enfants respectifs. Vous pouvez avoir plusieurs éléments Composite représentant différentes étapes ou branches du processus.

Un aspect à surveiller (et qui serait révélé par un échec dans la simulation) est d'avoir plusieurs threads qui traitent simultanément sur diagrammes distincts. La simulation ne peut pas passer à un nouveau diagramme si elle suit également un autre thread sur le diagramme actuel.

Ce diagramme donne un aperçu du processus de retrait d'espèces à un distributeur automatique de billets :



L'activité Retrait au DAB est un élément composite. Si vous double-cliquez dessus, vous ouvrez et affichez le diagramme enfant, qui est une analyse plus détaillée du processus de retrait. De même, une simulation ouvrira et traitera le diagramme enfant.



Simulation d'Interface Utilisateur Win32

Enterprise Architect supporte la simulation de boîtes de dialogue et d'écrans créés avec le profil Win32 @ Interface Utilisateur , pour intégrer la conception de l'interface utilisateur au comportement défini du système. Les boîtes de dialogue peuvent être référencées et invoquées par programmation à l'aide de commandes JavaScript dans un modèle comportemental tel qu'une Statemachine , offrant ainsi une exécution entièrement personnalisable et entièrement interactive de votre modèle comportemental.

Les boutons de commande peuvent être utilisés pour diffuser des signaux, en déclenchant un déclencheur lorsque le bouton est cliqué. Les arguments de signal peuvent être renseignés à partir des champs de saisie le dialogue , par exemple pour capturer et envoyer un nom d'utilisateur et un mot de passe pour évaluation.

Les boîtes de dialogue conçues à l'aide du profil Win32 Interface Utilisateur (et existant dans la même branche Paquetage que le modèle comportemental en cours d'exécution) seront créées en tant que nouvelles fenêtres en arrière-plan au début de la simulation. Diverses propriétés pouvant affecter l'apparence et le comportement de chaque dialogue et contrôle peuvent être personnalisées au moment de la conception via les Valeur Étiquetés fournies par le profil Win32 Interface Utilisateur .

Pour interagir avec un dialogue via JavaScript pendant la simulation, le mot clé de niveau simulation ' dialogue ' est utilisé, suivi d'un point et du nom le dialogue . Propriétés et méthodes sont alors accessibles ; par exemple, pour afficher le dialogue , ou pour définir la valeur de texte d'un 'Edit Control' :

```
dialog.Login.Show=true;
```

```
dialog.Login.Username.Text="admin";
```

Exemples

Pour visualiser un exemple de Simulation d'Interface Utilisateur Win32 , ouvrez le modèle EAExample et localisez le diagramme :

Exemple Modèle > Simulation de Modèle > Statemachine Models > Connexion client > Client > Connexion client

Propriétés communes

Ces propriétés et méthodes communes sont disponibles sur la plupart des types de contrôle UI Win32 pris en charge.

Propriété/Méthode	Description
Activer	Booléen Active ou désactive l'interaction de l'utilisateur.
Déplacer vers (x, y, largeur, hauteur)	Déplacez la fenêtre aux coordonnées spécifiées et définissez la hauteur et la largeur de la fenêtre.
Montrer	Booléen Afficher ou masquer le dialogue . Lorsque cette propriété est définie sur False, le dialogue est déplacé hors de l'écran.
Texte	String Définissez le titre de le dialogue ou de la fenêtre.

Fonctions JavaScript

Fonction	Description
BroadcastSignal (string Signal)	Envoie un signal à la file d'attente des événements de simulation. Paramètres: <ul style="list-style-type: none"> Signal : String – le nom du signal à diffuser
UIBroadcastSignal (string Signal, tableau Paramètres)	Envoie un signal avec des paramètres supplémentaires à la file d'attente des événements de simulation. Paramètres: <ul style="list-style-type: none"> Signal : String – le nom du signal à diffuser Paramètres : Tableau – paramètres supplémentaires à fournir pour ce signal Exemple: UIBroadcastSignal("Connexion",{Nom : dialogue .Login.Username.Text, Mot de passe : dialogue .Login.Password.Text});
ShowInterface (string InterfaceName, booléen Afficher)	Obsolète . Voir la propriété Show sur le contrôle « Dialogue ». Par exemple : dialogue .HelloWorld.Show = vrai;
InterfaceOperation (string InterfaceName, string ControlName, string OperationName,[string arg1],[string arg2])	Obsolète . Les opérations peuvent être référencées directement à partir du contrôle. Par exemple : dialogue .HelloWorld.ListControl.InsertItem(" Test ", 2);
GetInterfaceValue (string InterfaceName, string ControlName, string OperationName,[string arg1],[string arg2])	Obsolète . Propriétés peuvent être référencées directement à partir du contrôle. Par exemple : dialogue .HelloWorld.EditControl.Text;



Notes

- Les commandes doivent être dans un dialogue ; toutes les commandes en dehors d'un dialogue ne seront pas interprétées
- Les boîtes de dialogue et les contrôles doivent être sur une interface Win32 diagramme Interface Utilisateur
- Les contrôles UI simples et les contrôles UI de base peuvent également être utilisés dans une simulation, mais leurs fonctionnalités sont limitées par rapport aux contrôles UI Win32
- Les noms Dialogue et les noms de contrôle doivent être uniques ; si plusieurs contrôles du même nom existent, la simulation ne pourra pas les différencier
- Les espaces dans les noms dialogue et les noms de contrôle sont traités comme des traits de soulignement
- Les noms Dialogue et les noms de contrôle sont sensibles à la casse.

Contrôles UI Win32 pris en charge

Ce tableau identifie tous les contrôles UI Win32 disponibles dans Enterprise Architect pour la conception et la simulation de l'interface utilisateur.

Accéder

Ruban	Design > Diagramme > Toolbox :  > Spécifiez ' Interface Utilisateur - Win32' dans la dialogue ' Trouvez Item de Boîte à Outils '
Raccourcis Clavier	Ctrl+Shift+3 :  > Spécifiez ' Interface Utilisateur - Win32' dans la dialogue ' Trouvez Item de Boîte à Outils '

Contrôles UI Win32

Contrôle	Description
Bouton	<p>Les contrôles de bouton sont un moyen courant de permettre l'interaction de l'utilisateur pendant l'exécution ; par exemple, un bouton OK dans un écran de connexion. Un bouton peut répondre à un événement de clic, défini en ajoutant une Valeur Étiquetée « OnClick ».</p> <p>En réponse à un événement de clic, un bouton peut être utilisé pour, par exemple, envoyer un signal, provoquant le déclenchement d'un déclencheur pendant l'exécution.</p> <p>Propriétés de conception personnalisables :</p> <ul style="list-style-type: none"> • Client Edge • Bouton par défaut • Désactivé • Plat • Alignement horizontal • Cadre modal • Multiligne • Aligner le texte à droite • Ordre de lecture de droite à gauche • Bordure statique • Tabstop • Transparent • Alignement vertical • Visible <p>Valeur Étiquetés :</p> <ul style="list-style-type: none"> • OnClick – spécifie une commande JavaScript à exécuter en réponse à un événement de clic sur ce bouton

	<p>Propriétés :</p> <ul style="list-style-type: none"> • Activer • Montrer • Texte <p>Opérations :</p> <ul style="list-style-type: none"> • Déplacer vers
<p>Case à cocher</p>	<p>Propriétés de conception personnalisables :</p> <ul style="list-style-type: none"> • Auto • Client Edge • Désactivé • Plat • Alignement horizontal • Texte de gauche • Cadre modal • Multiligne • Aligner le texte à droite • Ordre de lecture de droite à gauche • Bordure statique • Tabstop • Alignement vertical • Visible <p>Valeur Étiquetés :</p> <ul style="list-style-type: none"> • OnCheck – spécifie une commande JavaScript à exécuter en réponse à un changement de la valeur de cette case à cocher <p>Propriétés :</p> <ul style="list-style-type: none"> • Checker – valeur integer [0 1] • Activer • Montrer • Texte
<p>Zone de liste déroulante</p>	<p>Propriétés de conception personnalisables :</p> <ul style="list-style-type: none"> • Auto • Client Edge • Données – string de valeurs délimitée par des points-virgules pour remplir la zone de liste déroulante au moment de l'exécution ; par exemple, « oui ; non ; peut-être » • Désactivé • A des cordes • Minuscule • Cadre modal • Aligner le texte à droite • Ordre de lecture de droite à gauche • Trier • Bordure statique • Tabstop

	<ul style="list-style-type: none"> • Type • Majuscule • Défilement vertical • Visible <p>Opérations</p> <ul style="list-style-type: none"> • AddString (string) • SupprimerTout() • DeleteItem (numéro) – supprimer l'élément à l'index spécifié • DeleteString (string) – supprime tous les éléments correspondant string • Obtenir le nombre () • GetString (nombre) • InsertItem (nombre, string) • InsertString (nombre, string) • SetString (nombre, string) <p>Propriétés :</p> <ul style="list-style-type: none"> • Activer • Sélection – index de l'élément actuellement sélectionné • Montrer
Dialogue	<p>Propriétés de conception personnalisables :</p> <ul style="list-style-type: none"> • Alignement absolu • Fenêtre d'application • Bordure - Redimensionnement ou cadre Dialogue uniquement • Centre • Client Edge • Centre de la souris • Clip Frères et sœurs • Désactivé • Barre de défilement horizontale • Barre de défilement gauche • Édition locale • Maximiser la boîte • Réduire la boîte • Pas d'activation • Fenêtre superposée • Fenêtre de palette • Aligner le texte à droite • Ordre de lecture de droite à gauche • Définir le premier plan • Menu système • Système modal • Barre de titre • Fenêtre d'outils • Le plus haut • Transparent

	<ul style="list-style-type: none"> • Barre de défilement verticale • Visible • Bordure de fenêtre Propriétés : <ul style="list-style-type: none"> • Activer • Montrer • Texte Opérations : <ul style="list-style-type: none"> • Déplacer vers
Contrôle d'édition / Contrôle d'édition enrichi	Propriétés de conception personnalisables <ul style="list-style-type: none"> • Aligner le texte • Défilement automatique HScroll • Défilement automatique • Frontière • Client Edge • Désactivé • Minuscules (contrôle d'édition uniquement) • Cadre modal • Multiligne • Nombre • Mot de passe • Lecture seule • Aligner le texte à droite • Ordre de lecture de droite à gauche • Bordure statique • Tabstop • Transparent • Majuscules (contrôle d'édition uniquement) • Visible • Je veux revenir Propriétés : <ul style="list-style-type: none"> • Activer • Montrer • Texte
Boîte de groupe	Propriétés de conception personnalisables : <ul style="list-style-type: none"> • Client Edge • Désactivé • Plat • Alignement horizontal • Cadre modal • Aligner le texte à droite • Bordure statique • Tabstop

	<ul style="list-style-type: none"> • Visible <p>Propriétés :</p> <ul style="list-style-type: none"> • Activer • Montrer • Texte
<p>Liste déroulante</p>	<p>Propriétés de conception personnalisables :</p> <ul style="list-style-type: none"> • Frontière • Client Edge • Désactiver le défilement nul • Désactivé • Barre de défilement gauche • Cadre modal • Aligner le texte à droite • Sélection • Trier • Bordure statique • Tabstop • Défilement vertical • Visible <p>Opérations :</p> <ul style="list-style-type: none"> • AddString (string) • SupprimerTout() • DeleteItem (numéro) – supprimer l'élément à l'index spécifié • DeleteString (string) – supprime tous les éléments correspondant string • Obtenir le nombre () • GetString (nombre) • InsertItem (nombre, string) • InsertString (nombre, string) • SetString (nombre, string) <p>Propriétés :</p> <ul style="list-style-type: none"> • Activer • Sélection – index de l'élément actuellement sélectionné • Montrer
<p>Contrôle de liste</p>	<p>Propriétés de conception personnalisables :</p> <ul style="list-style-type: none"> • Alignement • Toujours afficher la sélection • Frontière • Client Edge • Désactivé • Modifier les étiquettes • Barre de défilement gauche • Cadre modal • Pas d'en-tête de colonne

	<ul style="list-style-type: none"> • Pas de défilement • Sélection unique • Trier • Bordure statique • Tabstop • Vue • Visible <p>Valeur Étiquetés :</p> <ul style="list-style-type: none"> • Colonnes – string permettant d'initialiser les noms et les tailles des colonnes pour ce contrôle de liste, séparées par des points-virgules : par exemple, « Colonne1 ; 100 ; Colonne2 ; 150 ; » <p>Opérations :</p> <ul style="list-style-type: none"> • AddString (string) • SupprimerTout() • DeleteItem (numéro) – supprimer l'élément à l'index spécifié • DeleteString (string) – supprime tous les éléments correspondant à la string • Obtenir le nombre () • GetString (nombre, nombre) • InsertItem (nombre, string) • InsertString (nombre, string) • SetString (nombre, nombre, string) <p>Propriétés :</p> <ul style="list-style-type: none"> • Activer • Sélection – index de l'élément actuellement sélectionné • Montrer
<p>Contrôle de l'image</p>	<p>L'image initiale du Picture Control peut être définie à l'aide de la Valeur Étiquetée 'Image'. Définissez la valeur sur un nom de fichier accessible par la simulation. L'image peut être modifiée au moment de l'exécution à l'aide de la méthode ChangeImageFile en JavaScript . Cela prend un seul paramètre string du nom de fichier à charger.</p> <p>Définissez la propriété « Type d'image » sur le type de fichier approprié (Bitmap, Métafichier amélioré ou Icône). Ce paramètre ne peut pas être modifié lors de l'exécution.</p> <p>Propriétés de conception personnalisables :</p> <ul style="list-style-type: none"> • Frontière • Image centrale • Client Edge • Couleur (couleur du cadre) • Désactivé • Type d'image • Cadre modal • Image en taille réelle • Bordure statique • Tabstop • Vue • Visible

	<p>Opérations :</p> <ul style="list-style-type: none"> • ChangeImageFile (string) - nom de fichier <p>Propriétés :</p> <ul style="list-style-type: none"> • Montrer
<p>Contrôle de progression</p>	<p>Propriétés de conception personnalisables :</p> <ul style="list-style-type: none"> • Frontière • Client Edge • Désactivé • Chapiteau • Cadre modal • Lisse • Bordure statique • Tabstop • Verticale • Visible <p>Valeur Étiquetés :</p> <ul style="list-style-type: none"> • Plage – string spécifiant les valeurs minimales et maximales pour ce contrôle, séparées par un point-virgule : par exemple, « 1 ; 100 » <p>Propriétés :</p> <ul style="list-style-type: none"> • Activer • Pos • Gamme • Montrer • Étape
<p>Bouton radio</p>	<p>Propriétés de conception personnalisables :</p> <ul style="list-style-type: none"> • Auto • Client Edge • Désactivé • Plat • Groupe • Alignement horizontal • Texte de gauche • Cadre modal • Multiligne • Bordure statique • Tabstop • Alignement vertical • Visible <p>Valeur Étiquetés :</p> <ul style="list-style-type: none"> • OnChangeSelection – spécifie une commande JavaScript à exécuter en réponse à un changement de sélection de ce bouton radio <p>Propriétés :</p> <ul style="list-style-type: none"> • Checker – valeur integer [0 1] • Activer

	<ul style="list-style-type: none"> • Sélection – valeur integer • Montrer
<p>Contrôle du curseur</p>	<p>Propriétés de conception personnalisables :</p> <ul style="list-style-type: none"> • Coche automatique • Frontière • Client Edge • Désactivé • Activer la plage de sélection • Cadre modal • Orientation • Indiquer • Bordure statique • Tabstop • Coches • Transparent • Fond transparent • Info-bulles • Visible <p>Valeur Étiquetés :</p> <ul style="list-style-type: none"> • Plage – string spécifiant les valeurs minimales et maximales pour ce contrôle, séparées par un point-virgule : par exemple, « 1 ; 100 » <p>Propriétés :</p> <ul style="list-style-type: none"> • Activer • Taille de la page • Pos • Gamme • Montrer
<p>Contrôle de rotation</p>	<p>Propriétés de conception personnalisables :</p> <ul style="list-style-type: none"> • Alignement • Touches fléchées • Auto Buddy • Client Edge • Désactivé • Cadre modal • Pas de milliers • Orientation • Définir Integer Buddy • Bordure statique • Tabstop • Visible • Envelopper <p>Valeur Étiquetés :</p> <ul style="list-style-type: none"> • Plage – string spécifiant les valeurs minimales et maximales pour ce contrôle, séparées par un point-virgule : par exemple, « 1 ; 100 »

	<p>Propriétés :</p> <ul style="list-style-type: none"> • Activer • Pos • Gamme • Montrer
Texte statique / Étiquette	<p>Propriétés de conception personnalisables :</p> <ul style="list-style-type: none"> • Aligner le texte • Frontière • Client Edge • Désactivé • Fin des points de suspension • Cadre modal • Ellipse de chemin • Pas d'emballage • Notifier • Ellipse de chemin • Aligner le texte à droite • Simple • Bordure statique • Creux • Tabstop • Visible • Ellipse des mots <p>Propriétés :</p> <ul style="list-style-type: none"> • Activer • Montrer • Texte
Contrôle des onglets	<p>Propriétés de conception personnalisables :</p> <ul style="list-style-type: none"> • Frontière • Bas • Boutons • Client Edge • Désactivé • Boutons plats • Focus • Piste chaude • Cadre Modèle • Multiligne • Aligner le texte à droite • Bordure statique • Style • Tabstop • Info-bulles

	<ul style="list-style-type: none"> • Visible <p>Valeur Étiquetés :</p> <ul style="list-style-type: none"> • Onglets – string spécifiant les noms de chaque onglet pour ce contrôle, séparés par un point-virgule : par exemple, « Onglet 1 ; Onglet 2 ; Onglet 3 ; » <p>Propriétés :</p> <ul style="list-style-type: none"> • Activer • Montrer
Contrôle des arbres	<p>Propriétés de conception personnalisables :</p> <ul style="list-style-type: none"> • Toujours afficher la sélection • Frontière • Cases à cocher • Client Edge • Désactiver le glisser-déposer • Désactivé • Modifier les étiquettes • Sélection de ligne complète • A des boutons • A des lignes • Défilement horizontal • Barre de défilement gauche • Lignes à la racine • Cadre modal • Aligner le texte à droite • Ordre de lecture de droite à gauche • Rouleau • Simple Développer • Bordure statique • Tabstop • Info-bulles • Sélection de piste • Visible <p>Opérations :</p> <ul style="list-style-type: none"> • Delete() - supprime le TreeItem spécifié • InsertItem (string) - chemin en pointillé du nouvel élément d'arborescence à insérer ; tous les éléments parents de ce chemin en pointillé qui n'existent pas encore seront créés automatiquement • InsertString (string) - Voir <i>InsertItem</i> • TreeItem (string) - chemin en pointillés de l'élément d'arborescence auquel accéder ; utilisez la propriété <i>Text</i> pour définir le texte de cet élément d'arborescence ou utilisez l'opération <i>Delete</i> pour supprimer cet élément de l'arborescence <p>Propriétés :</p> <ul style="list-style-type: none"> • Activer • Sélection – string contenant le chemin en pointillés de l'élément d'arbre sélectionné • Montrer

- Texte – obtenir ou définir le texte d'un TreeItem spécifié

Exemples :

```
dialogue .MyDialog.MyTreeControl.InsertItem("Racine.Parent.Enfant");
```

```
dialogue .MyDialog.MyTreeControl.TreeItem("Root.Parent.Child").Text =  
"Modifié";
```

```
dialogue .MyDialog.MyTreeControl.Selection = "Racine.Parent";
```

```
dialogue .MyDialog.MyTreeControl.TreeItem("Root.Parent.Modified").Delete();
```

Win32 Control Valeur Étiquetés

Diverses propriétés pouvant affecter l'apparence et le comportement de chaque dialogue et contrôle Win32 peuvent être personnalisées au moment de la conception via Valeur Étiquetés fournies par le profil Interface Utilisateur Win32.

Valeur Étiquetés

Certains types de contrôles supportent l'ajout de Valeur Étiquetés spéciales qui modifient leur comportement.

Les contrôles tels que les boutons, les cases à cocher et les boutons radio peuvent réagir aux événements de l'interface graphique et exécuter une commande JavaScript. Pour permettre à un contrôle de répondre à un événement, créez une nouvelle Valeur Étiquetée avec un nom approprié, par exemple « OnClick », puis saisissez la commande JavaScript dans la valeur.

Les contrôles à onglets peuvent utiliser une Valeur Étiquetée « Onglets » pour définir les onglets qui apparaîtront dans ce contrôle lorsqu'il est simulé.

Les contrôles de curseur, les contrôles de rotation et les contrôles de progression peuvent utiliser une Valeur Étiquetée « Plage » pour définir les valeurs minimales et maximales par défaut acceptées par le contrôle pendant la simulation.

Étiquette	Description
Colonnes	<p>S'applique à : Contrôle de liste</p> <p>Utilisation : initialise les noms et largeurs de colonnes pour un contrôle de liste. Chaque nom et largeur de colonne sont séparés par un point-virgule ; par exemple, « Colonne1 ; 100 ; Colonne2 ; 150 ; " ».</p>
Sur clic	<p>S'applique à : Bouton</p> <p>Utilisation : identifie la commande JavaScript à exécuter en réponse à un événement de clic sur un contrôle Button.</p>
Survérification	<p>S'applique à : case à cocher</p> <p>Utilisation : identifie la commande JavaScript à exécuter en réponse à un changement de valeur d'un contrôle Check Box.</p>
Sélection sur le changement	<p>S'applique à : bouton radio</p> <p>Utilisation : identifie la commande JavaScript à exécuter en réponse à un changement de valeur d'un contrôle Radio Button.</p>
Gamme	<p>S'applique à : Contrôle du curseur, Contrôle de rotation, Contrôle de la progression</p> <p>Utilisation : spécifie les valeurs minimales et maximales par défaut du contrôle, séparées par un point-virgule : par exemple, « 1 ; 100 ».</p>
Onglets	<p>S'applique à : Contrôle des onglets</p> <p>Utilisation : spécifie le nom de chaque onglet à créer pour le contrôle d'onglets, séparé par un point-virgule : par exemple, « Onglet 1 ; Onglet 2 ; Onglet 3 ; " ».</p>

Simulation BPMN

La simulation BPMN est une méthode de visualisation et de validation du comportement de vos diagrammes BPMN Processus Métier . Grâce aux indications visuelles de toutes les activités en cours d'exécution et des activités possibles qui peuvent être exécutées ensuite, vous pourrez facilement identifier et résoudre les problèmes potentiels du processus que vous avez modélisé.

La simulation de modèles BPMN est similaire à la simulation de modèles Comportementale UML standard, sauf que BPMN :

- Utilise des types d'éléments différents (tels que Passerelle au lieu de Décision) et
- Fonctionne sur des scripts placés, généralement, dans le champ ' Valeur Étiquetée ' approprié associé aux connecteurs et éléments, au lieu des champs ' Propriétés ' (et, si vous préférez, plutôt que dans la dialogue ' Analyseur d'Exécution Build Scripts ') ; le script est écrit en JavaScript

Travailler avec Simulation BPMN

Activité	Détail
Créer un Modèle de Simulation BPMN	Lorsque vous créez un modèle BPMN adapté à la simulation, vous prenez en compte la manière dont vous représentez le point de départ, le flux et les conditions à tester.
Comparer les activités UML aux Processus BPMN	La simulation des modèles BPMN Processus Métier présente un certain nombre de différences avec la simulation des diagrammes d'activité UML .

Notes

- La simulation BPMN est disponible dans les éditions Unified et Ultimate d' Enterprise Architect

Créer un Modèle de Simulation BPMN

Dans le cadre du processus de développement d'un modèle de simulation, déterminez laquelle des trois options de réalisation de la simulation vous préférez appliquer :

- Exécutez un script de simulation pour initialiser les variables du diagramme - sélectionnez « BPMN » comme plate-forme, exécutez la simulation comme « En tant que script » et sélectionnez le script ; vous définirez ensuite les conditions et les décisions comme des déclarations JavaScript dans les Valeur Étiquetés des éléments et des connecteurs sur le diagramme , soit avant de démarrer la simulation, soit pendant la simulation
- N'utilisez pas de script, mais initialisez les variables dans la première activité et, encore une fois, modifiez les conditions et les décisions dans les Valeur Étiquetés des éléments et des connecteurs, puis exécutez la simulation comme « Interprétée » ; vous pouvez ensuite réinitialiser les variables pendant la simulation, ainsi que les conditions
- Exécutez la simulation en mode « Manuel » et gérez le flux et les conditions manuellement à chaque étape

Créer un diagramme BPMN adapté à la simulation

Étape	Action
1	Créez un diagramme Processus Métier ou BPEL à partir de la technologie BPMN 2.0. Si vous créez un diagramme BPEL, Enterprise Architect affiche des boîtes de dialogue spécialisées pour simplifier la création de modèles conformes.
2	Nous vous recommandons de créer un événement Démarrer pour indiquer clairement où démarre votre simulation. Vous avez plusieurs choix pour le Type d'événement ; le choix n'influence pas la simulation de votre modèle. Si aucun Événements Démarrer n'est défini, la simulation démarrera à partir d'une activité qui n'a pas de flux Séquence entrants.
3	Ajoutez toutes les activités impliquées dans le processus modélisé. Vous avez plusieurs choix pour le Type de tâche ; le choix n'influence pas la simulation de votre modèle. Le comportement des activités peut être décomposé davantage en spécifiant un Type d'activité de sous-processus et en sélectionnant Embedded ou CallActivity. Les boucles standard sont également prises en charge.
4	Ajoutez des flux Séquence entre vos activités. Dans la dialogue « Propriétés BPEL », vous pouvez saisir la condition qui doit être satisfaite (True) avant que la Flux séquence ne soit suivie. Vous pouvez également définir le type de condition sur « Par défaut » pour garantir que ce flux sera pris si toutes les autres branches échouent à la condition spécifiée. Si vous ne travaillez pas avec un diagramme BPEL, vous utilisez les valeurs conditionExpression et conditionType Valeur Étiquetés .
5	Ajoutez Événements de fin pour toutes les conditions qui provoqueront la fin du processus ou du chemin d'exécution actif. Vous avez plusieurs choix pour le Type d'événement ; parmi ceux-ci, seul le type Terminate influencera l'exécution. Dans les simulations avec plusieurs nœuds actifs, cela provoque la fin de l'ensemble du processus au lieu de seulement du thread qui atteint ce nœud.

Notes

- Pour inclure des activités qui se trouvent dans Paquetages externes au Paquetage simulé, dessinez un :
 - Connecteur Paquetage Import depuis le Paquetage contenant le diagramme étant simulé pour chaque Paquetage externe, ou
 - Connecteur de dépendance du Paquetage contenant le diagramme

étant simulé pour chaque activité dans les Paquetages externes

Initialiser Variables et Conditions

Pour un modèle de simulation BPMN, vous pouvez initialiser vos variables dans un script Analyseur d'Exécution . Vous pouvez également initialiser ces variables dans les Valeur Étiquetées du premier élément Activité du processus, ce qui vous donne une plus grande flexibilité pour ajouter et modifier des variables au fur et à mesure de la simulation. De même, vous pouvez définir les conditions et les valeurs à appliquer aux différents points de décision (Gateways) du processus, dans les Valeur Étiquetées des connecteurs Flux séquence .

Si vous souhaitez intégrer une interface utilisateur dans votre processus de simulation, en utilisant Win32, vous utilisez à nouveau Valeur Étiquetées pour identifier le dialogue ou prompt à afficher, dans l'élément Activité juste avant le point auquel la valeur ou la décision est traitée.

Pour la simulation de diagrammes UML , les variables à l'intérieur de l' object « sim » et object « this » sont affichées dans la fenêtre Variables locales.

Accéder

Affichez l'onglet 'Tags' de la fenêtre Propriétés en utilisant l'une des méthodes décrites ici.


Ruban	Explorer > Portails > Windows > Propriétés > Propriétés > Étiquettes
Raccourcis Clavier	Ctrl+2 > onglet 'Tags' de la fenêtre Propriétés

Initialiser les variables

1. Sur le diagramme , cliquez sur le premier élément Activité du processus.
2. Dans l'onglet 'Tags' de la fenêtre Propriétés , cliquez sur la flèche déroulante du champ « valeur » de taskType et sélectionnez « Script ».
3. Dans le champ « valeur » du script, saisissez le code JavaScript approprié, tel que :

```
sim.loan=true; sim.status="undefined";
```

Définir les conditions

1. Sur le diagramme , cliquez sur un connecteur Flux séquence issu d'un élément Passerelle .
2. Dans l'onglet 'Tags' de la fenêtre Propriétés , cliquez sur la flèche déroulante du champ « Valeur » de conditionType et sélectionnez « Expression ».
3. Dans le champ « Valeur » de conditionExpression (<mémo>*), cliquez sur le bouton  pour afficher la fenêtre Note Valeur Étiquetée . Type le code JavaScript approprié, par exemple :
sim.status=="Maintenir"
4. Cliquez sur le bouton OK . Le texte de l'instruction s'affiche comme étiquette du connecteur.

Incorporer Interface Utilisateur Win32

1. Sur le diagramme , cliquez sur l'élément Activité qui représente l'endroit où la décision est prise.

2. Dans l'onglet 'Tags' de la fenêtre Propriétés , cliquez sur la flèche déroulante du champ « taskType valeur » et sélectionnez « Script ».
3. Dans le champ « script valeur », saisissez le code JavaScript approprié, tel que :
dialogue .Screen1.Show=Vrai;
(Cette instruction affiche le dialogue Screen1. Vous pouvez masquer temporairement le dialogue en changeant « Afficher » sur False.)

Comparaison des activités UML et Processus BPMN

L'exécution et la simulation de modèles BPMN présentent un certain nombre de différences par rapport à l'exécution et à la simulation de diagrammes d'activité UML . La mise en correspondance de concepts similaires et les différences entre les deux méthodes d'expression du comportement d'un système sont présentées ici.

Comparaison des activités UML et Processus BPMN

Activité UML	BPMN Processus Métier
Le point de départ est défini par un nœud initial. Aucune méthode permettant de spécifier la raison pour laquelle l'activité a été démarrée n'est disponible.	Le point de départ est défini par un événement Démarrer . Cela implique une cause spécifique pour le démarrage de l'activité, bien qu'elle puisse être non spécifiée.
L'unité de comportement de base d'une activité est l'élément Action . UML fournit de nombreuses formes différentes d'actions, bien que la simulation n'utilise qu'un petit sous-ensemble de celles-ci.	L'unité de comportement de base d'une activité est l'élément Activité. Plusieurs types de tâches différents sont disponibles. Ceux-ci décrivent généralement différentes méthodes d'exécution (par exemple, manuelle) par opposition à ce qui se passe.
Un flux de contrôle est utilisé pour connecter les éléments d'un diagramme d'activité. Une fonctionnalité distinctive est qu'un seul flux de contrôle peut être suivi à partir de n'importe quel nœud, à l'exception d'un nœud de fourche explicite. Pour restreindre le flux sur un flux de contrôle, ajoutez une garde.	Une Flux séquence permet de relier les éléments sur un diagramme Processus Métier . Ceux-ci diffèrent des diagrammes d'activité UML dans la mesure où tous les flux de séquence valides sont pris par défaut. Pour restreindre le flux sur une Flux séquence définissez la conditionType Valeur Étiquetée sur 'Expression' et créez le script dans la conditionExpression Valeur Étiquetée .
Un nœud Décision est utilisé pour modéliser explicitement une décision en cours de prise. Un nœud Merge, qui utilise la même syntaxe, est utilisé lorsque les flux potentiels sont combinés en un seul.	Un nœud Passerelle défini sur « Exclusif » est utilisé lorsqu'un seul chemin doit être sélectionné. Il est également utilisé pour combiner à nouveau les flux potentiels. Une direction peut être spécifiée comme « Convergente » ou « Divergente » pour sélectionner explicitement entre les deux modes.
Un nœud Fork est utilisé pour exécuter simultanément plusieurs nœuds, tandis qu'un nœud	Un nœud Passerelle défini sur « Parallèle » est utilisé pour modéliser explicitement l'exécution simultanée de plusieurs nœuds. Il est également utilisé pour attendre que tous les flux entrants soient disponibles et repartir avec un seul flux. Une direction peut être spécifiée comme « Convergente » ou « Divergente » pour sélectionner

Join, utilisant la même syntaxe, est utilisé pour attendre que tous les flux entrants soient disponibles et partir avec un seul flux.	explicitement entre les deux modes.
Il n'est pas possible d'exécuter simultanément uniquement certaines sorties d'un nœud pour les activités UML . Si vous en avez besoin, ajoutez ultérieurement des flux de contrôle avec les protections appropriées.	Un nœud Passerelle défini sur Inclusive est utilisé pour modéliser explicitement la situation dans laquelle tous les flux sortants avec une condition vraie sont exécutés simultanément.
Une Action de comportement d'appel est utilisée lorsque le comportement doit être davantage décomposé en faisant référence à une activité externe.	Les éléments d'activité sont définis comme un sous-processus CallActivity lorsque le comportement doit être davantage décomposé en faisant référence à une activité externe.
Activité Action Appel Comportement Action .	Les éléments d'activité sont définis comme un sous-processus intégré lorsque le comportement doit être davantage décomposé sans faire référence à une activité externe.

