



ENTERPRISE ARCHITECT

Série de Guides d'Utilisateur

Analyse d'Exécution Visuelle

Author: Sparx Systems & Stephen Maguire

Date: 7/11/2024

Version: 17.0

CRÉÉ AVEC  **ENTERPRISE
ARCHITECT**

Table des Matières

Introduction	6
Analyse d'exécution	7
Construire et Déboguer	11
Scripts d'Analyseur	13
Gestion Scripts d'Analyseur	15
Éditeur de Script Analyseur	19
Créer Scripts	24
Script Nettoyage	26
Scripts de Test	28
Sortie des Testpoints	30
Script Débogage	32
Exigences Spécifiques au Système d'Exploitation	33
Systèmes d'Exploitation compatibles UAC	34
Débogage WINE	35
Java	37
Configuration générale pour Java	38
Techniques avancées	40
Se connecter à Virtual Machine	41
Applet Java Navigateur Internet	42
Travailler avec Serveurs Web Java	43
Serveur JBOSS	45
Serveur Apache Tomcat	46
Apache Tomcat Windows Service	47
.NET	48
Configuration générale pour .NET	49
Débogage d'une application non gérée	51
Déboguer COM Interop	52
Déboguer ASP .NET	53
Le Débogueur Mono	54
Configuration de débogage Linux	55
Fenêtres de configuration de Windows	57
Le Débogueur PHP	59
Débogueur PHP - Exigences système	62
Liste de contrôle Débogueur PHP	63
Le Débogueur GNU (GDB)	65
Le Débogueur Android	67
Débogueur Java JDWP	70
Sortie Point de Trace	73
Configuration Établi	74
Microsoft C++ et natif (C, VB)	75
Configuration générale	76
Symboles Débogueur	78
Script de fusion	79
Script Code Miner	80
Script de services	83
Exécuter le script	84
Script de déploiement	85

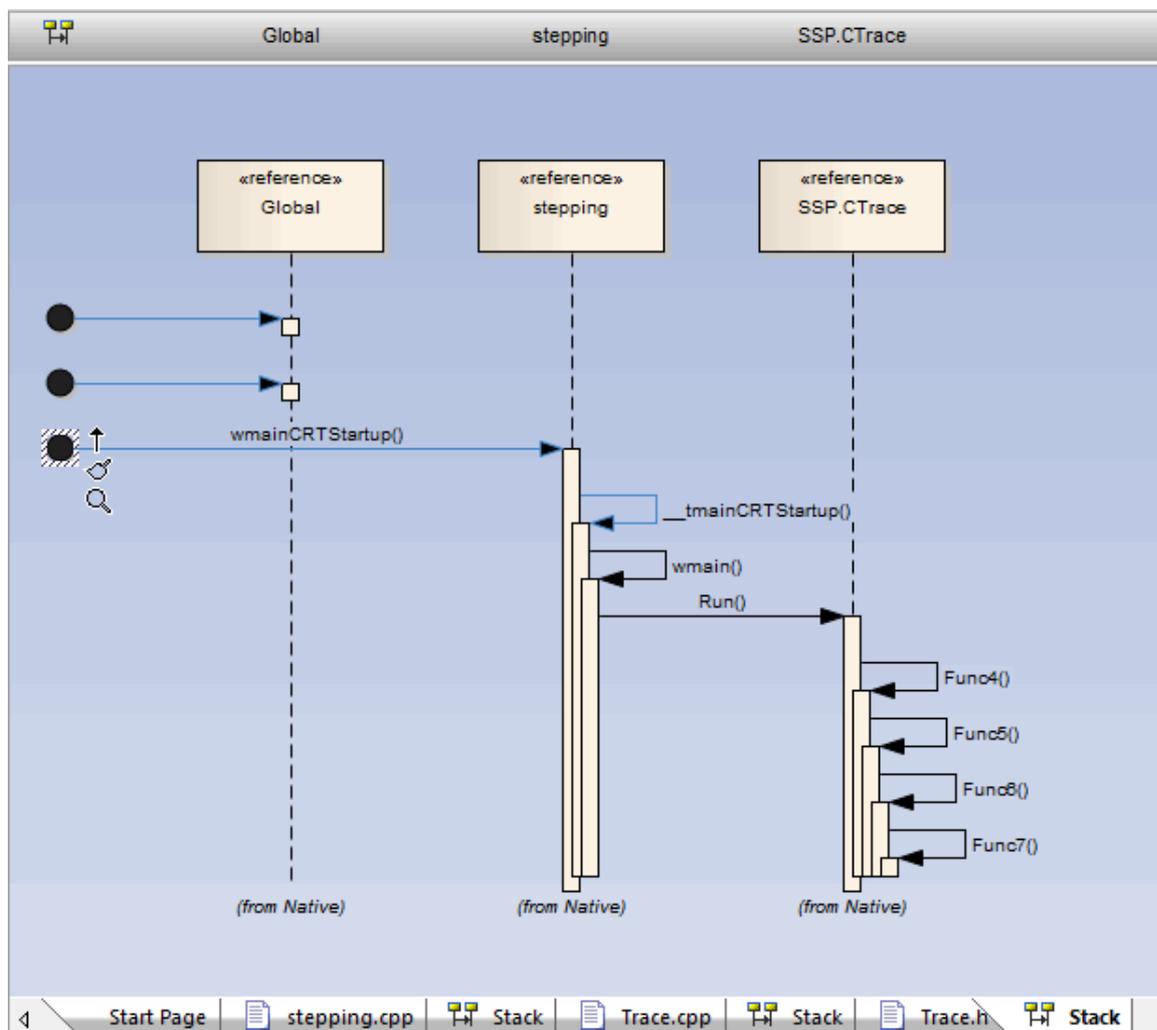
Scripts Enregistrement	87
La fenêtre de la file d'attente des tâches	89
Créer une application	93
Localiser les erreurs Compilateur dans le code	94
Débogage	95
Exécuter le Débogueur	97
Gestion des Point d'Arrêt et Marqueurs	100
Définition Points d'Arrêt du code	103
Déclarations de trace	104
Interrompre lorsqu'une Variable Change de Valeur	106
Trace lorsque Variable Change de Valeur	109
Détection des Opérations d'Adresse Mémoire	110
Point d'Arrêt Propriétés	112
Défaut de lier Point d'Arrêt	114
Déboguer une Application en Cours	115
Voir les Variables Locales	116
Voir le Contenu de Longues Chaînes	119
Variables Vue Déboguer dans Éditeurs de Code	121
Variables Instantanés	122
Points d'Action	124
Voir Variables dans d'Autres Portées	128
Voir Éléments du Réseau	129
Voir la Pile d'Appel	130
Créer Diagramme de Séquence de Pile d'Appel	132
Inspecter Mémoire de Processus	134
Afficher Modules Chargés	135
Traiter Exceptions à Première Chance	136
Débogueur juste à temps	138
Services	139
Fenêtre des services de l'analyseur	143
Enregistrement	145
Comment ça marche	149
L' Historique d'Enregistrement	151
Diagramme Fonctionnalités	153
Configuration pour Enregistrement	154
Contrôle Profondeur de Pile	155
Placer des Marqueurs d'Enregistrement	156
Coffret Marqueurs d'Enregistrement	157
Types de Marqueurs	158
La Fenêtres Points d'Arrêt et Marqueurs	160
Travailler avec Ensembles de Marqueurs	161
Contrôler la Session d'Enregistrement	163
Barre d'Outils d'Enregistreur	164
Travailler avec Historique d'Enregistrement	166
Démarrer Enregistrement	168
Parcourez Appels de Fonction	170
Marqueurs d'Enregistrement Nested	171
Génération de Diagrammes Séquence	172
Rapportage les transitions State	174
Rapportage d'une Statemachine	175
Enregistrement et Mappage Changements de State	177

Analyseur de State	179
Synchronisation	186
Visualiser State Exécuter	188
Object Établi	189
Utilisant l'Établi	190
Créer des objets	191
Invocation Méthodes	193
Définition Propriétés	195
Débogage et Établi	196
Enregistrement et l' Établi	197
Suppression d'objets	199
Fermeture de l' Établi	200
Profilage	201
Exigences du système	208
Démarrage	209
Graphique d'Appel	211
Profil de Pile	214
Profil de mémoire	216
Fuites de Mémoire	218
Options de Réglage	221
Démarrer et Arrêter le Profileur	223
Rapports de Ligne de Fonction	225
Générer, Enregistrer et Charger des Rapports de Profil	228
Enregistrer Rapport dans Bibliothèque d'Équipe	233
Testpoints	234
Diagramme Domaine de Test	238
Coupe Test	240
Ensemble de Test	241
Suite de Test	242
La Fenêtre Testpoints	243
Barre d'Outils Testpoints	245
Éditeur Testpoints	247
Contraintes Testpoint	249
Test Unitaire	252
Configurer Tests Unitaires	253
Exécuter Tests Unitaires	255
Enregistrer Résultats des Tests	256
Échantillons	257

Introduction

Enterprise Architect, in addition to its extensive features as a fully fledged Integrated Development Environment, provides tools for visualizing and analyzing software execution that have a profound effect on the way that code is managed, maintained and documented. Much of this capability is unique to Enterprise Architect, which uses its rich visualization and modeling facilities to bring programming code to life, effectively allowing the code to be put under the microscope.

Code hot spots and faults can be analyzed, errors and opportunities for speed improvements can be identified quickly, and resolutions to errors found. These features can be used while building and testing software products, but are also profoundly useful for documenting existing or legacy systems. Commonly there is no documentation for these systems and it is prohibitively expensive or even intractable to get the code documented by conventional means. Enterprise Architect can create sophisticated, elegant and valuable models and documents that completely describe the code base, including expressive Sequence diagrams and static models. This type of documentation is a number of generations more advanced than the simplistic and often opaque documentation generation tools such as Javadoc.



Analyse d'exécution



L' Analyseur d'Exécution Visuelle (VEA) est composé d'une suite d'outils avancés et complets qui vous permettent de créer, déboguer, enregistrer, profiler, simuler et de construire et vérifier votre développement logiciel tout en gardant le code étroitement intégré à votre modèle. Enterprise Architect prend support une large gamme de compilateurs et de plates-formes populaires, en particulier les environnements Java, .Net et Microsoft Windows C++. Le développement logiciel devient une expérience visuelle hautement rationalisée, très différente du travail dans des environnements traditionnels.

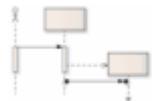
Enterprise Architect est lui-même modélisé, construit, compilé, débogué, testé, géré, profilé et autrement construit entièrement au sein de l' Analyseur d'Exécution Visuelle intégré à Enterprise Architect . Bien que le VEA puisse être utilisé pour compléter d'autres suites d'outils, il brille également lorsqu'il est utilisé comme IDE de développement principal en couplage étroit avec les capacités de gestion de modèles et de projets fournies par Enterprise Architect .

Accéder

Ruban	Develop > Source Code > Analyseur d'Exécution > Edit Scripts d'Analyseur Exécuter > Outils > Analyseur
Raccourcis Clavier	Maj+F12

Analyseur d'Exécution Fonctionnalités

Fonctionnalité	Description
Construire et Déboguer 	En utilisant Scripts d'Analyseur liés aux Modèle Paquetages , il est possible d'intégrer étroitement le cycle de code/construction/débogage dans Enterprise Architect . Pour Java, .Net et Microsoft C++ en particulier, il est simple de se lier aux bases de code du projet et de prendre en charge la construction et le débogage dans l'environnement de développement piloté Modèle d' Enterprise Architect . En plus fonctionnalités de débogage standard, le couplage fort avec le modèle et l'utilisation de fonctionnalités de débogage avancées telles que les points Action font Enterprise Architect la plate-forme idéale à partir de laquelle concevoir et construire votre application logicielle.
Simulation 	Donnez 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 de simulation, ainsi qu'à la possibilité de suivre visuellement l'exécution au moment de l'exécution, le simulateur est un moyen efficace de « regarder les roues tourner » en visualisant

	l'exécution de vos modèles comportementaux.
<p>Profilage</p> 	Soulevez le capot des performances logicielles et voyez ce qui se passe réellement. Obtenez rapidement une image claire des raisons pour lesquelles certaines tâches se comportent mal ou plus mal que prévu. Qu'il s'agisse de Microsoft .NET , de C++ natif ou de Java, utilisez des profils pour évaluer efficacement les changements de performances au cours du cycle de vie de votre logiciel.
<p>Enregistrement Exécution</p> 	Enregistrez l'exécution du code sans avoir recours à l'instrumentation. Contrôlez la quantité de détails grâce aux filtres et à la profondeur de la pile. Générer de magnifiques diagrammes Séquence et diagrammes qui illustrent la collaboration de classe. Utilisez l'enregistrement pour créer diagrammes de domaine Test qui peuvent être utilisés avec la fonctionnalité VEA Testpoints .
<p>Tester</p> 	Créez et gérez des scripts de test pour les éléments du modèle. Explorez l'interface Tester , qui prend en charge les tests unitaires, d'intégration, de scénario, de système, d'inspection et d'acceptation. Utilisez la méthodologie de programmation par contrat avec le facilité Testpoints .
<p>Object Établi</p> 	Comportement de la classe Établi à la volée, en les instanciant dans l' Object Établi puis en appelant leurs opérations. Vous pouvez même transmettre des objets sur l'établi en tant que paramètres à d'autres objets de l'établi.
<p>Analyseur d'Exécution Visuelle Samples</p> 	Essayez nos exemples motifs pour configurer et explorer le riche ensemble de fonctionnalités de l' Analyseur d'Exécution Visuelle .

Avantages de l' Analyseur d'Exécution

L' Analyseur d'Exécution fournit un environnement de développement et de test intégré pour plusieurs plates-formes, notamment Microsoft .NET , Java, Native C++, Mono et Android. Il comprend un débogueur riche en fonctionnalités, l'enregistrement et le profilage des exécutions et la gestion Testpoint .

Il vous aide à générer diagrammes Séquence , de classe de domaine Test et de classe collaborative à partir d'un seul enregistrement. C'est un excellent moyen de comprendre et de documenter votre application.

- Visualiser l'exécution du programme
- Optimiser les ressources système existantes et comprendre l'allocation des ressources
- Vérifiez que le système suit les règles comme prévu
- Produire une documentation de haute qualité qui reflète plus précisément le comportement du système
- Comprendre comment et pourquoi les systèmes fonctionnent
- Former les nouveaux employés à la structure et au fonctionnement d'un système
- Fournir une compréhension complète du fonctionnement du code existant
- Identifier les appels de fonctions coûteux ou inutiles
- Illustrer les interactions, les structures de données et les relations importantes au sein d'un système
- Tracer les problèmes jusqu'à une ligne de code spécifique, une interaction système ou un événement

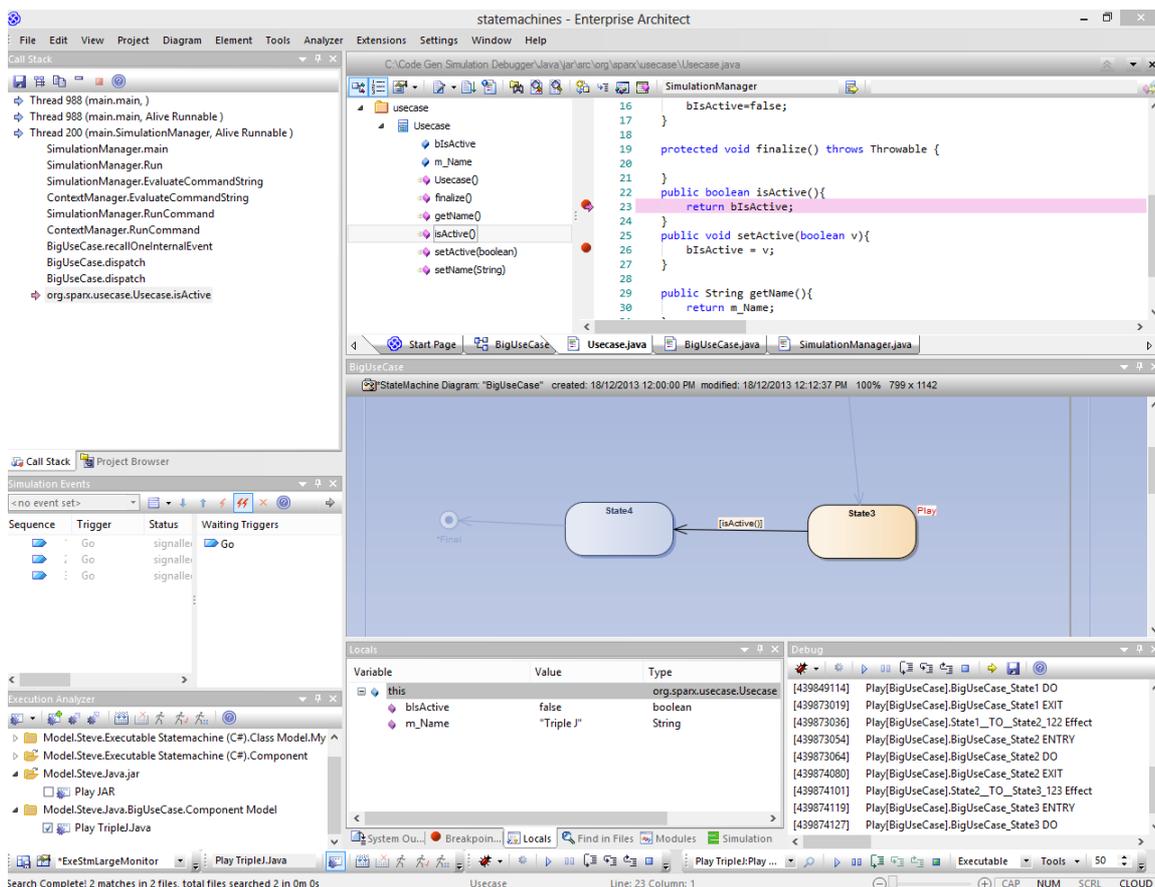
- Établir la séquence des événements qui se produisent immédiatement avant la défaillance du système
- Simuler l'exécution de modèles de comportement, y compris Statemachines , les activités et les interactions

Opérations

Opération	Description
Simuler le comportement	<p>Simulez des modèles de comportement UML pour vérifier leur exactitude logique et de conception, pour :</p> <ul style="list-style-type: none"> • Activités • Interactions et séquences • Statemachines
Exécution d'enregistrement	<p>Enregistrer les programmes en cours d'exécution et représenter le comportement sous forme de diagramme Séquence UML ; l'enregistrement est pris en charge pour :</p> <ul style="list-style-type: none"> • Microsoft Windows natif C, C++, Visual Basic • Famille Microsoft .NET (C# , J#, VB) • Java • Mono • Androïde • PHP
Comportement du profil	<p>Affichez/créez rapidement des rapports sur les comportements des applications en cours d'exécution. Le profilage est pris en charge pour les plates-formes suivantes :</p> <ul style="list-style-type: none"> • Microsoft C natif, C++, Visual Basic • Famille Microsoft .NET (C# , J#, VB) (y compris tout mélange de code non géré/géré) • Java • Mono
Test les cas d'utilisation avec le système Testpoints	<p>La gestion Testpoints facilité de définir les contraintes sur un modèle de classe sous forme de contrats. Les contrats fournissent les ressources sur lesquelles créer des domaines Test . Un seul domaine Testpoint peut ensuite être utilisé pour tester et signaler le comportement de plusieurs applications. Vous pouvez également utiliser l' Analyseur d'Exécution pour enregistrer un cas d'utilisation et générer un diagramme de domaine Test avec très peu d'effort. Tous Testpoints existants sont automatiquement liés au domaine généré ou le diagramme de domaine Test peut être utilisé comme contexte pour de nouvelles compositions de contrat. Le comportement d'une application pour un domaine Test donné peut être vu immédiatement en temps réel ! Les résultats sont affichés dans la fenêtre de rapport de Testpoint chaque fois qu'un contrat est réussi ou échoué. Le découplage de la mesure de test de la base de code présente un certain nombre d'avantages, dont l'un est de faciliter la réconciliation de plusieurs systèmes avec un domaine Test commun, plutôt que l'un de l'autre.</p> <p>Le système Testpoint supporte ces contrats :</p> <ul style="list-style-type: none"> • Invariants de classe • Conditions préalables de la méthode • Post-conditions de la méthode

	<ul style="list-style-type: none">• État des lignes
Ouvrir la fenêtre de la console	La fenêtre Console est un interpréteur de ligne de commande grâce auquel vous pouvez rapidement créer une fenêtre de terminal pour activer un moteur de script et entrer des commandes pour agir sur le script (JScript, JavaScript et VBScript).
Object Établi	Créez et travaillez avec des objets créés dans l'environnement modélisation Enterprise Architect à l'aide d'un Object Établi dynamique. <ul style="list-style-type: none">• Créer des objets à partir du modèle de classe• Appeler des méthodes et afficher les résultats• Collaboration de classe Établi• Transmettre des objets en tant que paramètres à d'autres objets• fonctionnalités de débogage complètes, y compris l'enregistrement
Exécuter des tests xUnit	Exécuter des tests NUnit et JUnit pour Java et Microsoft .NET Enregistrer et documenter les résultats.
Importer le code à partir de l'analyse	Enregistrement et le profilage d'exécution acquièrent tous deux une collection de fichiers de code pertinents, que vous pouvez rétroconcevoir vers le modèle actuel en une seule opération.

Construire et Débuguer



Enterprise Architect s'appuie sur ses capacités exceptionnelles de génération de code, de création de diagrammes et de conception avec une suite complète d'outils permettant de créer, déboguer, visualiser, enregistrer, tester, profiler et de construire et vérifier des applications logicielles. L'ensemble d'outils est intimement lié aux capacités modélisation et de conception et fournit un moyen unique et pratique de construire un logiciel à partir d'un modèle et de maintenir la synchronisation entre le modèle et le code.

Enterprise Architect vous aide à définir des « Scripts d'Analyseur » liés aux Paquetages Modèle qui décrivent comment une application sera compilée, quel débogueur utiliser et d'autres informations connexes telles que les commandes de simulation. Le script d'analyseur est l'élément de configuration principal qui relie votre code aux fonctionnalités de création, de débogage, de test, de profilage et de déploiement dans Enterprise Architect .

Pour mesurer la compétence de l'ensemble d'outils, il convient de noter qu'Enterprise Enterprise Architect est en fait entièrement construit, débogué, profilé, testé et développé dans l'environnement de développement Enterprise Architect . De nombreux outils de débogage avancés tels Action Points ont été développés pour résoudre les problèmes inhérents à la construction d'applications logicielles volumineuses et complexes (telles qu'Enterprise Enterprise Architect) et sont utilisés quotidiennement par l'équipe de développement Sparx Systems .

Il est recommandé aux nouveaux utilisateurs de prendre le temps de bien comprendre l'utilisation des Scripts d'Analyseur et la manière dont ils lient le modèle au code, aux compilateurs et aux autres outils nécessaires à la construction de logiciels.

Intégration Modèle et du code

L'ingénierie pilotée par Modèle est une approche moderne du développement logiciel qui promet une productivité accrue et un code de meilleure qualité, ce qui permet aux systèmes d'être commercialisés plus rapidement et avec moins de défauts. Ce qui rend cette approche convaincante est la possibilité de décrire et de maintenir l' architecture et la conception d'un système dans un modèle, puis de générer du code de programmation et des schémas qui peuvent être

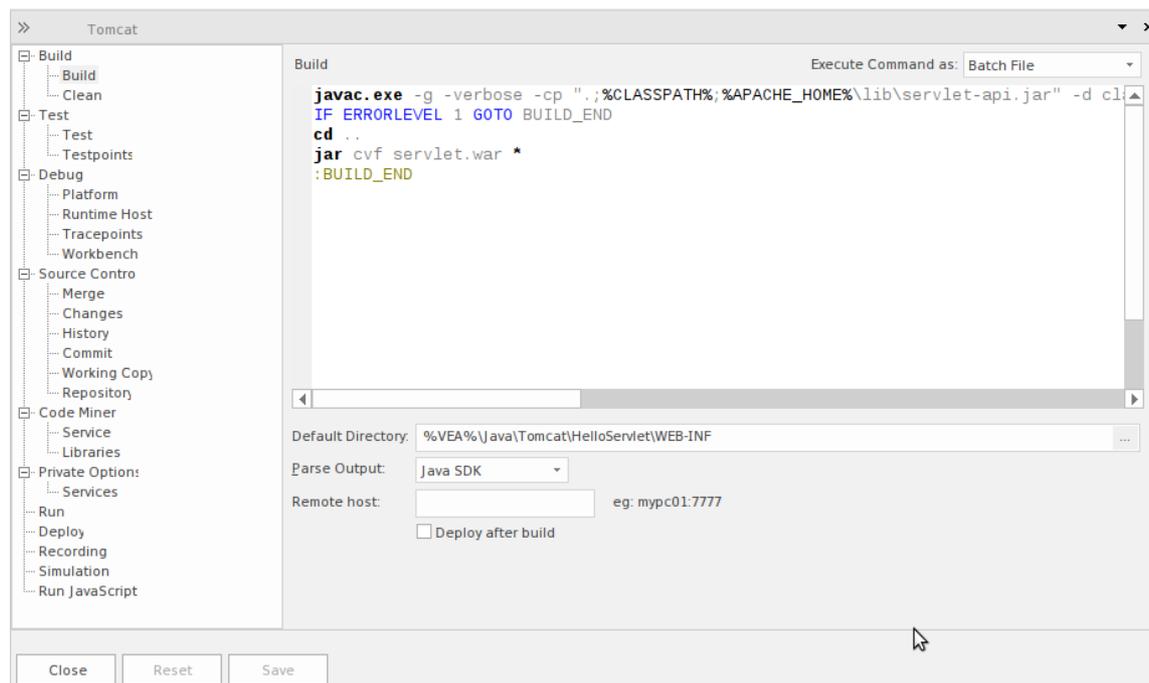
synchronisés et visualisés dans le modèle.

L'environnement de développement piloté par Modèle (MDDE) d' Enterprise Architect supporte cette approche et fournit un ensemble d'outils flexibles pour augmenter la productivité et réduire les erreurs. Il s'agit notamment de la possibilité de définir l'architecture et la conception dans des modèles, de générer du code à partir de ces modèles, de synchroniser le code avec les modèles et de maintenir le code dans des éditeurs de code sophistiqués. Le code source ou les binaires peuvent également être importés, et les utilisateurs peuvent enregistrer et documenter le code préexistant ou récemment développé. L'outil Analyzer Script vous aide à décrire comment créer, déboguer, tester et déployer une application.

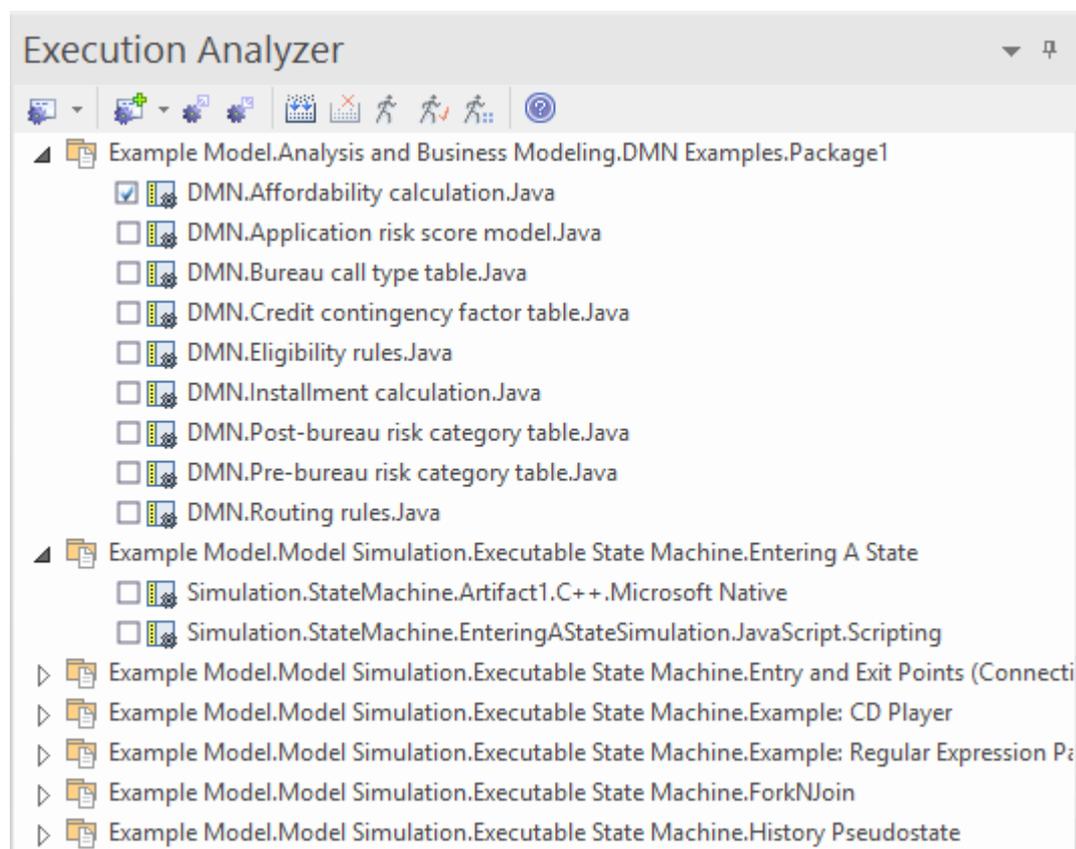
Facilité	Description
<p>Développement piloté par Modèle</p>	<p>Le développement piloté Modèle offre un cycle de développement plus robuste, accessible et plus rapide que les cycles traditionnels axés sur le codage.</p> <p>Un modèle bien construit, intimement lié aux capacités de création, exécuter, de débogage, de test et de déploiement du code source, fournit une architecture cible riche, facile à parcourir et à comprendre. La traçabilité, le lien avec les cas d'utilisation, les composants et d'autres artefacts du modèle, ainsi que la capacité d'enregistrer et de documenter facilement le code préexistant ou récemment développé, rendent l'environnement de développement d' Enterprise Architect particulièrement efficace.</p> <p>Enterprise Architect intègre des langages d'édition, de débogage et modélisation intelligents standard de l'industrie.</p>
<p>L'environnement de développement piloté par Modèle (MDDE)</p>	<p>Le MDDE fournit des outils pour concevoir, visualiser, construire et déboguer une application :</p> <ul style="list-style-type: none"> • Technologies et outils UML pour modéliser les logiciels • Outils de génération de code pour générer/rétroconcevoir le code source • Outils pour importer du code source et des binaires • Éditeurs de code support différents langages de programmation • Intelli-sense pour faciliter le codage • Scripts d'analyse qui permettent à un utilisateur de décrire comment créer, déboguer, tester et déployer l'application • Intégration avec des compilateurs tels que Java, .Net, Microsoft C++ • Capacités de débogage pour Java, .NET, Microsoft C++ et autres • Capacités avancées de visualisation, d'enregistrement, d'inspection, de test et de profilage <pre data-bbox="523 1413 1406 1662"> pApp = new CBCGPAAppointmentDemo (COleDateTime& dtStart, COleDateTime& dtFinish, CString& strText, COLORREF clrBackground, COLORREF clrForeground, COLORREF clrDuration); </pre>

Scripts d'Analyseur

Scripts d'Analyseur sont utilisés par l'Analyseur d'Exécution. Vous n'avez pas à vous soucier de leur création. Ils ne sont pas du même type de script que JavaScript ou PHP, mais sont gérés à l'aide d'une interface utilisateur familière - une arborescence - et vous pouvez rapidement localiser la fonctionnalité à modifier. Scripts d'Analyseur peuvent être partagés par les utilisateurs d'un modèle communautaire et sont facilement importés et exportés sous forme de fichiers XML.



Un même projet peut avoir plusieurs configurations et celles-ci peuvent être regroupées dans la fenêtre de l'analyseur.



Chaque script d'analyse est défini pour un Paquetage , de sorte que les projets peuvent coexister en toute tranquillité. Dans de nombreuses organisations, les procédures de gestion des systèmes sont distribuées et varient d'un individu à l'autre et d'un groupe à l'autre. Scripts d'Analyseur dans un modèle Enterprise Architect peuvent apporter une certaine tranquillité d'esprit à ces organisations, en faisant confiance à une procédure unique, partagée et responsable pour la création et le déploiement de toute sorte de configurations. Tous les aspects d'un script sont facultatifs. Vous pouvez, par exemple, déboguer sans un ; cependant, avec quelques lignes, ils peuvent activer ces fonctionnalités utiles :

- Bâtiment
- Tester
- Débogage
- Enregistrement
- Exécution
- Déploiement
- Simulation

Exécution de script à distance

Plusieurs sections de script Analyzer, telles que Build et Exécuter , fournissent un champ « Hôte distant ». Ce champ est utilisé pour décrire l'ordinateur sur lequel le script doit exécuter . Pour utiliser cette fonctionnalité , le service Sparx Satellite doit être exécuté sur la machine. Le format de ce champ est *hostname:port*, où *hostname* est l'adresse IP ou le nom de réseau d'une machine Windows ou Linux et *port* est le numéro de port sur lequel le service Satellite écoute. L'objectif principal de cette fonctionnalité est de permettre à un utilisateur d' Enterprise Architect exécuté sur Linux d'exécuter des commandes natives de Linux.

Gestion Scripts d'Analyseur

La fenêtre Analyseur d'Exécution vous permet de gérer tous les scripts Analyzer du modèle. Vous pouvez utiliser les boutons de la barre d'outils de la fenêtre ou les options du menu contextuel des scripts pour contrôler les tâches des scripts. Scripts sont répertoriés par Paquetage ; la liste ne montre que Paquetages pour lesquels des scripts Analyzer sont définis. Chaque utilisateur peut définir son propre script actif, indépendamment des autres utilisateurs du même modèle ; l'activation d'un script par un utilisateur n'a pas d'impact sur les scripts actuellement actifs des autres utilisateurs ni sur les scripts qui leur sont accessibles. Le script actif régit le comportement de l'Analyseur d'Exécution ; lors du choix de la commande de construction dans un menu, par exemple, ou lors du clic sur le bouton Déboguer dans une barre d'outils.

Il peut y avoir de nombreux Paquetages avec des scripts répertoriés dans la fenêtre. Pour vous aider à localiser et isoler un Paquetage particulier, utilisez les options du menu contextuel « Filtrer Paquetages », décrites dans le tableau *Options Menu Contexte* de cette rubrique.

Note que lorsqu'un script est sélectionné dans la fenêtre Analyseur d'Exécution, les options du menu contextuel de la fenêtre et les boutons de la barre d'outils de la fenêtre Analyseur d'Exécution s'appliqueront au script sélectionné. Cependant, les options Analyseur d'Exécution sur n'importe quel ruban ou barre d'outils flottante, ou dans le Débogueur (qui ne sont pas dans la fenêtre) utiliseront toujours le script Analyseur par défaut - celui dont la case à cocher est sélectionnée à côté du nom du script.

Accéder

Ruban	Develop > Source Code > Analyseur d'Exécution > Edit Scripts d'Analyseur Exécuter > Outils > Analyseur
Raccourcis Clavier	Maj+F12

Options de la barre d'outils

Bouton de la barre d'outils	Action
	Accès rapide aux fenêtres principales de l'Analyzer telles que Pile d'Appel ou Variables Locales, ainsi qu'aux puissantes fonctionnalités : <ul style="list-style-type: none"> • Profilage • Enregistrement • Testpoints • Simulation
	Créez et modifiez un nouveau script d'analyse pour le Paquetage sélectionné, sous Linux ou Windows™.
	Exporter Scripts . Exportez un ou plusieurs Scripts d'Analyseur vers un fichier XML, qui peut être utilisé pour importer les scripts dans un autre modèle. La dialogue « Analyseur d'Exécution : Exporter » s'affiche, à partir de laquelle vous sélectionnez le ou les scripts à exporter, suivi d'une prompt pour le nom et l'emplacement du fichier cible.

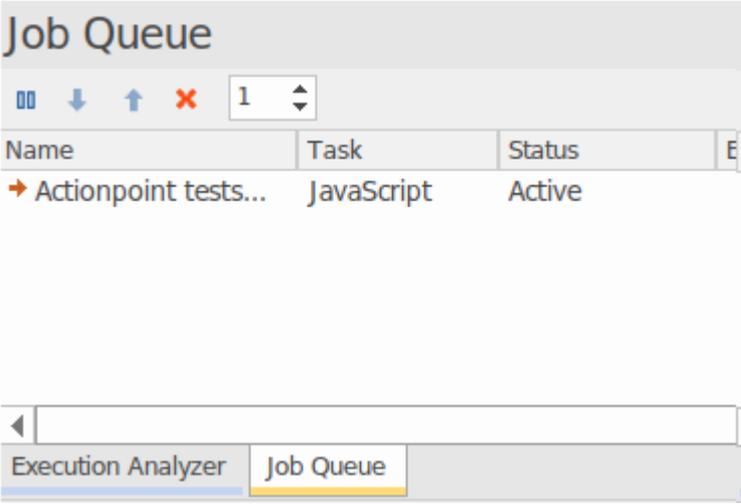
	<p>Importer Scripts .</p> <p>Importez un ou plusieurs Scripts d'Analyseur dans le modèle courant à partir d'un fichier XML préalablement exporté.</p> <p>La dialogue « Rechercher Paquetage » s'affiche. Vous pouvez y sélectionner le Paquetage dans lequel importer les scripts, puis prompt le nom du fichier source et son emplacement.</p>
	Exécutez la commande « Build » du script actif.
	Annuler la commande « Build » actuellement en cours.
	Exécutez la commande ' Exécuter ' du script actif.
	Exécutez la commande « Test » du script actif.
	Exécutez la commande « Déployer » du script actif.

Options Menu Contexte

Cliquez-droit sur le script ou Paquetage souhaité pour afficher les menus contextuels.

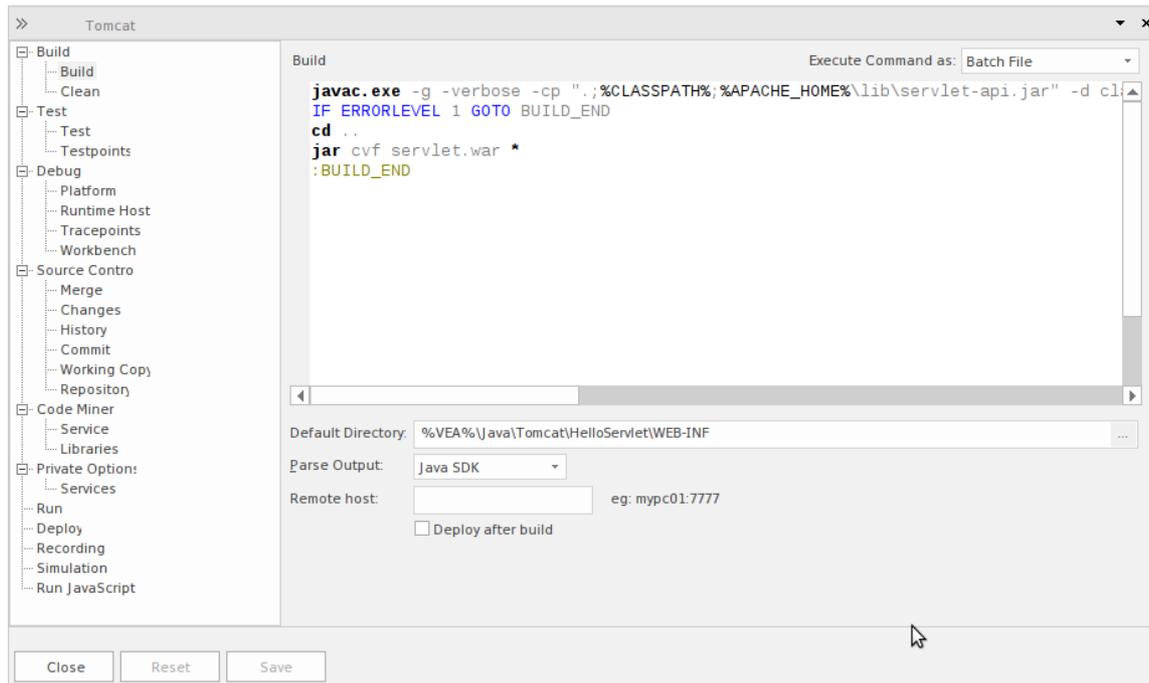
Option	Action
Ajouter un nouveau script	<p>Ajouter un nouveau script au Paquetage sélectionné.</p> <p>La fenêtre Analyseur d'Exécution s'affiche, montrant la page « Build ».</p>
Coller le script	<p>Collez un script copié depuis le presse-papiers Enterprise Architect dans le Paquetage sélectionné.</p> <p>Vous pouvez coller le script copié plusieurs fois ; chaque copie porte le suffixe « Copie ».</p> <p>Pour renommer le script copié, appuyez sur F2 et écrasez le nom du script.</p>
Exporter Scripts	<p>Exporter les scripts du Paquetage sélectionné.</p> <p>La dialogue « Analyseur d'Exécution : Exporter » s'affiche, à partir de laquelle vous sélectionnez le ou les scripts à exporter, suivi d'une prompt pour le nom du fichier cible et l'emplacement.</p>
Importer Scripts	<p>Importez des scripts depuis un fichier .XML dans le Paquetage sélectionné.</p> <p>Une prompt s'affiche pour le nom du fichier source et l'emplacement.</p>
Paquetages de filtres	<p>Affiche un sous-menu d'options pour :</p> <ul style="list-style-type: none"> Entrez le chemin Paquetage par rapport auquel filtrer la liste des Paquetages - lorsque vous sélectionnez l'option « Filtrer Paquetages », une prompt s'affiche pour accepter le chemin Paquetage actuellement sélectionné, ou vous pouvez supprimer les éléments <i>de fin</i> du chemin pour spécifier un ensemble plus grand de Paquetages ; lorsque vous cliquez sur le bouton OK , le premier Paquetage de la liste est développé pour répertorier les scripts qu'il contient Basculer entre l'affichage de la liste complète des Paquetages et le masquage de

	<p>la liste complète des Paquetages pour exposer uniquement le Paquetage actuellement sélectionné</p> <ul style="list-style-type: none"> • Supprimez le filtre actuellement actif pour afficher la liste complète des Paquetages
Sélectionner dans Navigateur de projet	Mettez en surbrillance le Paquetage sélectionné dans la fenêtre Navigateur . Affichez la fenêtre Navigateur , qui est maintenant agrandie pour afficher le Paquetage en surbrillance.
Construire	Exécutez la commande « Build » du script sélectionné.
Faire le ménage	Exécutez la commande « Nettoyer » du script sélectionné.
Reconstruire	Exécutez les commandes « Nettoyer » et « Construire » du script sélectionné.
Déboguer	Exécutez la commande ' Déboguer ' du script sélectionné.
Exécuter	Exécutez la commande ' Exécuter ' du script sélectionné.
Test	Exécutez la commande « Test » du script sélectionné.
Déployer	Exécutez la commande « Déployer » du script sélectionné.
Fusionner	Exécutez la commande de contrôle de source « Fusionner » du script sélectionné.
Changements	Exécutez la commande de contrôle de source « Modifications » du script sélectionné.
Histoire	Exécutez la commande de contrôle de source « Historique » du script sélectionné.
Committre	Exécutez la commande de contrôle de source « Commit » du script sélectionné.
Copie de travail	Exécutez la commande de contrôle de source « Copie de travail » du script sélectionné.
Référentiel	Exécutez la commande de contrôle de source ' Référentiel ' du script sélectionné.
Exécuter JavaScript	Exécutez la commande ' Exécuter JavaScript ' du script sélectionné. Lorsque cette option est sélectionnée, un job est créé dans la fenêtre Job Queue.

	
Démarrer Simulation	Démarrer la simulation référencée par la page ' Simulation de script d'analyse'.
Exécuter Exécutable Statemachine	Démarrer une simulation de l'artefact Statemachine exécutable sélectionné.
Modifier	Ouvrir le script sélectionné dans l'éditeur Scripts d'Analyseur .
Copie	Copiez le script sélectionné dans le presse-papiers Enterprise Architect .
Coller	<p>Collez le script le plus récemment copié dans le même Paquetage que le script sélectionné.</p> <p>Vous pouvez coller le script copié plusieurs fois ; chaque copie porte le suffixe « Copie ».</p> <p>Pour renommer le script copié, appuyez sur F2 et écrasez le nom du script.</p>
Supprimer	<p>Supprimez le script sélectionné ; aucune prompt de confirmation n'est affichée.</p> <p>Pour supprimer un Paquetage de la fenêtre de Analyseur d'Exécution , supprimez les scripts du Paquetage . Lorsque le dernier script est supprimé, le Paquetage n'est plus répertorié.</p>
Paquetage par défaut	<p>Définissez le script sélectionné comme script par défaut pour le Paquetage .</p> <p>L'icône à gauche du script change de couleur ; tout Paquetage par défaut précédent revient à la normale.</p>

Éditeur de Script Analyseur

L'Éditeur de Script Analyseur dispose d'une interface utilisateur simple, avec une arborescence des scripts sur la gauche permettant une navigation facile dans les groupes de scripts, et une vue de contenu sur la droite dans laquelle vous définissez et configurez les scripts.



Accéder

Depuis la fenêtre ' Analyseur d'Exécution ', soit :

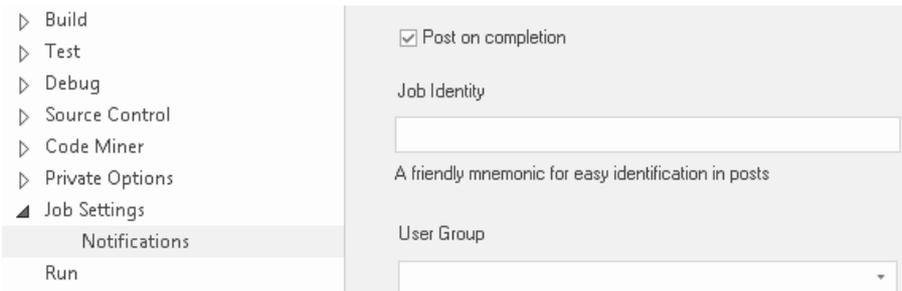
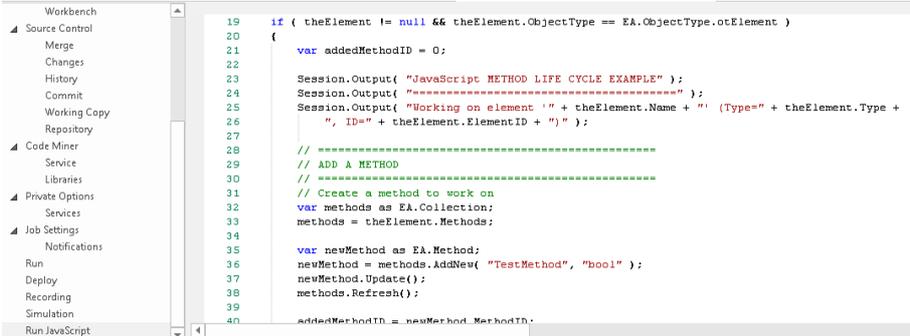
- Double-cliquez sur un script pour le modifier ou
- Cliquez-droit sur un script et sélectionnez l'option 'Modifier'

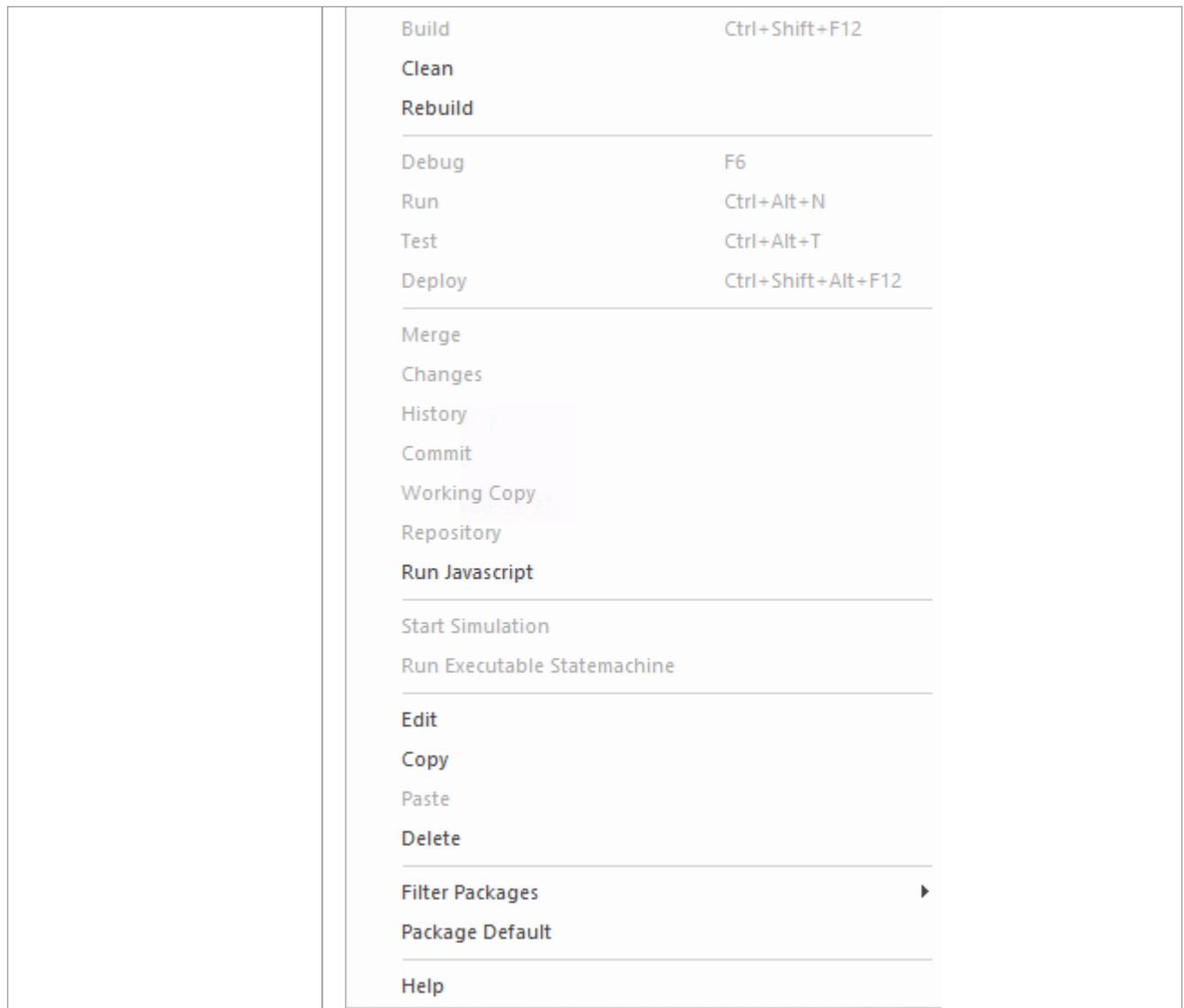
Ruban	Develop > Source Code > Analyseur d'Exécution > Edit Scripts d'Analyseur Exécuter > Outils > Analyseur
Raccourcis Clavier	Maj+F12

Analyseur d'Exécution Scripts

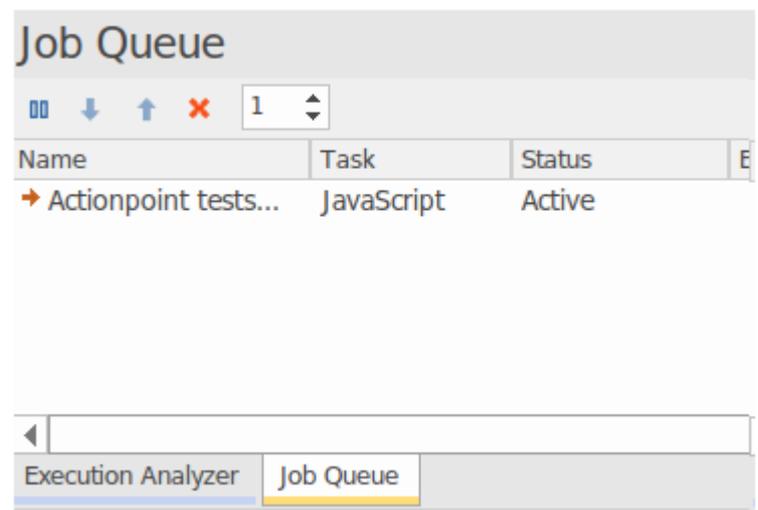
Tâche - Page	Action
Construire - Construire	Saisissez le script ou la commande pour générer l'application. Il peut s'agir d'une commande Apache Ant ou Visual Studio, mais elle peut également être personnalisée en fonction de votre environnement de développement. Note : n'oubliez pas de sélectionner un analyseur pour accéder directement au code source en cas d'erreur. Le champ Analyseur se trouve sur la même page et offre support de nombreux langages.

Construire - Nettoyer	Saisissez le script ou la commande pour nettoyer la version précédente. Il s'agit de la ligne de commande que vous devez normalement exécuter pour créer votre système. Il peut s'agir d'une commande Apache Ant ou Visual Studio en fonction de votre environnement de développement.
Test - Test	Entrez le script ou la commande pour tester l'application. C'est généralement à cet endroit qu'une invocation NUnit ou JUnit peut être configurée, mais cela pourrait tout aussi bien être n'importe quelle procédure ou programme.
Test - Testpoints	Spécifiez où la sortie d'un exécuteur Testpoint est envoyée.
Déboguer - Plateforme	Spécifiez la plateforme de débogage, l'application à déboguer et le mode de débogage (attacher au processus ou exécuter).
Déboguer - Tracepoints	Spécifiez où la sortie des points de trace rencontrés lors d'une session de débogage est envoyée.
Déboguer - Établi	Pour les projets .NET, l'assembly à charger. Non requis pour Java.
Déboguer - Hôte d'exécution	Permet à Enterprise Architect de lancer le programme à déboguer à l'aide d'une ligne de commande. Ceci est généralement utilisé pour les programmes Mono ou Java qui utilisent un transport de socket pour le débogage. Cette commande est exécutée avant que le Débogueur ne soit exécuté. Le numéro de port spécifié dans cette commande doit être la même valeur transmise à l'option « Port » dans la page Déboguer. Lorsque le Débogueur démarre, il tente de se connecter à l'environnement d'exécution sur ce port. En cas de succès, il lie ensuite tous les points d'arrêt et reprend le programme, qu'il suppose suspendu. Java et Mono ont tous deux des options de ligne de commande sur le transport de débogage pour suspendre initialement le processus jusqu'à ce que le Débogueur se connecte.
Contrôle de la source - Fusion	Il s'agit du script qui s'exécute lorsque l'option « Fusionner » est choisie dans le menu contextuel d'un script Analyzer. Il fournit un emplacement pour exécuter un programme ou un script shell afin d'examiner les différences entre les fichiers sources.
Contrôle des sources – Modifications	Il s'agit du script qui s'exécute lorsque l'option « Modifications » est sélectionnée dans le menu contextuel d'un script Analyzer. Il fournit un emplacement pour exécuter un programme de contrôle de source tel que « svn » qui peut répertorier les modifications actuelles apportées à un référentiel de contrôle de source.
Contrôle des sources – Historique	Il s'agit du script qui s'exécute lorsque l'option « Historique » est sélectionnée dans le menu contextuel d'un script Analyzer. Il fournit un emplacement pour exécuter un programme de contrôle de source tel que « svn » qui peut répertorier un historique des modifications apportées à un référentiel de contrôle de source.
Contrôle de la source - Validation	Il s'agit du script qui s'exécute lorsque l'option « Valider » est choisie dans le menu contextuel d'un script Analyzer. Il fournit un emplacement pour exécuter un programme de contrôle de source tel que « svn » qui peut valider les modifications apportées à une copie de travail de contrôle de source.
Contrôle de la source - Copie de travail	Il s'agit du script qui s'exécute lorsque l'option « Copie de travail » est sélectionnée dans le menu contextuel d'un script Analyzer. Il fournit un emplacement pour exécuter un programme de contrôle de source tel que « svn » pour effectuer des actions sur la copie de travail actuelle d'un référentiel source.

<p>Contrôle des sources - Référentiel</p>	<p>Il s'agit du script qui s'exécute lorsque l'option ' Référentiel ' est choisie dans le menu contextuel d'un script Analyzer. Il fournit un endroit où exécuter un programme de contrôle de source tel que 'svn' pour effectuer des actions sur un référentiel source.</p>
<p>Code Miner - Service</p>	<p>Dans cette section, vous pouvez choisir le mode de fonctionnement du service Code Miner . Vous pouvez choisir d'utiliser un serveur distant ou d'utiliser des bibliothèques localement.</p>
<p>Code Miner – Bibliothèques</p>	<p>Cette section fournit un espace pour la gestion des bibliothèques Code Miner . Ici, vous pouvez créer des bibliothèques basées sur une base de code ou un référentiel de projet. Les bibliothèques Code Miner créées ici peuvent être recherchées à l'aide de requêtes mFQL. Les requêtes composées en mFQL peuvent être utilisées pour rechercher une ou plusieurs bibliothèques en une seule opération.</p>
<p>Options privées - Services</p>	<p>C'est ici que sont configurés l'adresse IP et le port des services Enterprise Architect Satellite pour Linux et Windows . Ces services fournissent support à l'échelle de l'entreprise pour les fonctions de gestion du système et les scénarios de débogage à distance.</p>
<p>Paramètres de travail</p>	<p>La plupart des commandes contenues dans un script Analyzer sont exécutées sous forme de tâches dans la file d'attente des tâches. Chaque script peut être configuré pour envoyer une notification à un groupe d'utilisateurs spécifié lorsqu'une tâche spécifiée est terminée. Le groupe « Paramètres de tâche » fournit l'option « Notifications », qui affiche les champs via lesquels vous saisissez l'identité de la tâche dans le cadre du texte à afficher aux membres du groupe d'utilisateurs Mail de Modèle , ainsi que le nom du groupe d'utilisateurs.</p>  <ul style="list-style-type: none"> • Publier à l'achèvement - cochez cette case pour activer les deux autres champs • Identité du travail - Type le texte à afficher, qui doit également identifier le travail ; par exemple : « Installateurs terminés » • Groupe d'utilisateurs - Cliquez sur la flèche déroulante et sélectionnez le groupe d'utilisateurs Mail de Modèle approprié
<p>Exécuter JavaScript</p>	<p>Dans cette section, vous pouvez créer et stocker un JavaScript que vous pouvez exécuter en sélectionnant l'option « Exécuter JavaScript » dans le menu contextuel d'un script Analyzer.</p> 



Lorsque cette option est sélectionnée, un travail est créé dans la fenêtre File d'attente des travaux.



Exécuter	Entrez une commande pour exécuter une application.
	Entrez un script ou une commande pour déployer le projet. Créez votre fichier jar.

Déployer	Déployez sur votre appareil, un émulateur ou un serveur Tomcat. Publiez un site Web. C'est à vous de décider.
Enregistrement	Votre diagramme Séquence ressemble-t-il à la grille nationale ? Réduisez l'encombrement avec des filtres. Les filtres définissent des zones d'exclusion dans votre base de code qui peuvent réduire considérablement tout « bruit » enregistré. Même un bruit précis n'est pas toujours utile.
Simulation	Terminez la configuration pour le contrôle Simulation .

Créer Scripts

La page « Build » vous permet de saisir des commandes pour créer votre projet. Vous pouvez utiliser les chemins locaux et les variables d'environnement Enterprise Architect pour composer vos lignes de commande. Vous pouvez choisir de créer votre propre script de création en saisissant diverses commandes shell. Vous pouvez également choisir d'exécuter simplement un programme externe ou un fichier de commandes tel qu'un script Ant.

Accéder

Dans la fenêtre Analyseur d'Exécution , soit :

- Recherchez et double-cliquez sur le script requis et sélectionnez la page « Build > Build » ou
- Cliquez sur  dans la barre d'outils de la fenêtre, sélectionnez le Paquetage dans lequel créer un nouveau script et sélectionnez la page « Build > Build »

Ruban	Develop > Source Code > Analyseur d'Exécution > Edit Scripts d'Analyseur Exécuter > Outils > Analyseur
Menu Contexte l'analyseur	Construire
Raccourcis Clavier	Ctrl+Maj+F12

Construire

Écrivez votre script dans cette zone de texte, en utilisant les commandes du shell Windows ; le format et le contenu de cette section dépendent du compilateur que vous utilisez pour créer votre projet. Si le chemin ou les arguments contiennent des espaces, entourez-les de guillemets ; par exemple : « c:\program files (x86)\java\bin\javac.exe ».

Voici quelques exemples :

Visual Studio :

```
"C:\Program Files (x86)\Microsoft Visual Studio 9.0\Common7\IDE\devenv.com" /Rebuild Déboguer RentalSystem.sln
```

Visual Studio utilisant un chemin local :

```
"%VsCompPath%\devenv.exe" /build Déboguer Subway.sln
```

Java:

```
C:\Program Files (x86)\Java\jdk1.6.0_22\bin\javac.exe" -g -cp "%classpath%;. " %r*.java
```

Java utilisant un chemin local :

```
"%JAVA%\bin\javac.exe" -g -cp "%classpath%;. " %r*.java
```

Génération Java génériques (%r) - Les fichiers sources dans les sous-dossiers peuvent être générés à l'aide du jeton %r. Le jeton a pour effet de provoquer une exécution récursive de la même commande sur tous les fichiers de tous les sous-dossiers, comme illustré dans l'exemple.

Exécuter la commande comme

Cliquez sur la flèche déroulante et sélectionnez le mode d'exécution du script :

- Fichier batch - Utilisez cette option pour exécuter un script shell dans une fenêtre de commande système ; les variables d'environnement sont accessibles par les commandes de ce script
- Processus - Utilisez cette option pour exécuter la commande en tant que programme unique ; la commande doit spécifier le chemin d'accès au programme, ainsi que tous les arguments de ligne de commande

Répertoire par défaut

Recherchez ou saisissez le chemin du répertoire par défaut dans lequel le processus de script de génération sera exécuter .

Analyser la sortie

Cliquez sur la flèche déroulante et sélectionnez une méthode pour analyser automatiquement la sortie du compilateur. Si vous sélectionnez cette option, la sortie du script est enregistrée dans la fenêtre Sortie système ; Enterprise Architect analyse la sortie selon la syntaxe que vous spécifiez.

Hôte distant

Type l' ID du système hôte distant et son port ; par exemple, mypc01:7777.

Si vous définissez cette propriété sur #SYSTEMHOST#, le script est envoyé au Service Satellite Windows lorsqu'il est exécuté sous Windows et au Service Satellite Linux lorsqu'il est exécuté sous Wine. Les ID de service et les ports sont définis dans la section « Options privées - Services » de l'éditeur Scripts d'Analyseur .

Déployer après la construction

Cochez cette case pour que le script de déploiement soit exécuté immédiatement après la fin de ce script de build.

Notes

Pour exécuter le script Build, cliquez sur le Paquetage dans la fenêtre Navigateur et soit :

- Cliquez-droit sur n'importe quelle barre d'outils et sélectionnez 'Analyser Barres d'Outils | Construire', ou
- Appuyez sur Ctrl+Maj+F12 ou
- Sélectionnez l'option de ruban « Exécuter > Source > Créer > Créer »

Script Nettoyage

Les builds incrémentielles consistent à ne construire que les ressources qui ont changé d'une manière ou d'une autre. Il arrive cependant que l'on doive tout reconstruire à partir de zéro. Cette commande est utilisée dans ces cas-là, pour supprimer les binaires et les fichiers intermédiaires associés à une build ou une configuration particulière. Le projet peut alors être reconstruit. Lorsque vous exécutez l'option de menu « Rebuild » sur un script, la ou les commandes que vous spécifiez dans ce champ sont exécutées, suivies immédiatement de la commande « Build » du même script Analyzer. Certains compilateurs ont des options qui le font pour vous. Visual Studio par exemple dispose du commutateur de ligne de commande " /clean ».

Ceci est un exemple de script : devenv.com /Clean Déboguer MyProject.sln

Accéder

Dans la fenêtre Analyseur d'Exécution , soit :

- Recherchez et double-cliquez sur le script requis et sélectionnez la page « Build > Clean » ou
- Cliquez sur  dans la barre d'outils de la fenêtre, sélectionnez le Paquetage dans lequel créer un nouveau script et sélectionnez la page « Build > Clean »

Ruban	Develop > Source Code > Analyseur d'Exécution > Edit Scripts d'Analyseur Exécuter > Source > Créer > Nettoyer
Menu Contexte l'analyseur	Faire le ménage
Raccourcis Clavier	Maj+F12

Aspects

Aspect	Détail
Faire le ménage	Saisissez la commande à exécuter lorsque vous sélectionnez « Nettoyer » dans le menu contextuel du script.
Exécuter la commande comme	<p>Cliquez sur la flèche déroulante et sélectionnez l'option appropriée :</p> <ul style="list-style-type: none"> • Fichier de commandes - Utilisez cette option pour créer un script shell qui est exécuté dans une fenêtre de commande système ; les variables d'environnement sont accessibles par les commandes de ce script • Processus - Utilisez cette option pour exécuter un seul programme - la commande doit spécifier le chemin d'accès au programme, ainsi que tous les arguments de la ligne de commande ; - si le chemin ou les arguments contiennent des espaces, entourez-les de guillemets - par exemple : "c:\program files (x86)\java\bin\javac.exe"
Répertoire par défaut	La valeur par défaut est la valeur entrée pour le script de construction. Si aucune valeur n'a été définie pour le script de construction, recherchez ou saisissez le chemin du répertoire par défaut dans lequel le processus de script de nettoyage sera exécuter .

Analyser la sortie	<p>Cliquez sur la flèche déroulante et sélectionnez une méthode pour analyser automatiquement la sortie du compilateur.</p> <p>Si vous sélectionnez cette option, la sortie du script est enregistrée dans la fenêtre Sortie système ; Enterprise Architect analyse la sortie selon la syntaxe que vous spécifiez.</p>
Hôte distant	<p>Type l' ID du système hôte distant et son port ; par exemple, mypc01:7777.</p> <p>Si vous définissez cette propriété sur #SYSTEMHOST#, le script est envoyé au Service Satellite Windows lorsqu'il est exécuté sous Windows et au Service Satellite Linux lorsqu'il est exécuté sous Wine. Les ID de service et les ports sont définis dans la section « Options privées - Services » de l'éditeur Scripts d'Analyseur .</p>

Scripts de Test

Ces sections expliquent comment configurer la page « Test » d'un script Analyzer pour effectuer des tests unitaires sur votre code. La plupart des utilisateurs appliqueront cela aux scénarios de test NUnit et JUnit. Enterprise Architect accepte la sortie de ces systèmes et peut automatiquement ajouter et gérer l'historique de chaque cas de test unitaire. Pour afficher l'historique du cas, sélectionnez l'élément Classe du cas de test et appuyez sur Alt+2 > Tester .

Accéder

Dans la fenêtre Analyseur d'Exécution , soit :

- Recherchez et double-cliquez sur le script requis et sélectionnez la page « Test > Test » ou
- Cliquez sur  dans la barre d'outils de la fenêtre, sélectionnez le Paquetage dans lequel créer un nouveau script et sélectionnez la page « Test > Test »

Ruban	Develop > Source Code > Analyseur d'Exécution > Edit Scripts d'Analyseur Exécuter > Outils > Analyseur
Menu Contexte l'analyseur	Test
Raccourcis Clavier	Maj+F12

Actes

Test Type la commande ou le script Test dans ce champ. Par exemple :

- NUnit - "C:\Program Files\NUnit\bin\nunit-console.exe"
"bin\debug\Calculator.exe"
- JUnit - java junit.textui.Testrunner %N

La commande répertoriée dans ce champ est exécutée comme si elle provenait de l'invite de commande ; par conséquent, si le chemin de l'exécutable ou des arguments contiennent des espaces, ils doivent être entourés de guillemets.

Si vous incluez la string %N dans votre script de test, elle est remplacée par le nom entièrement qualifié de l'espace de noms de la classe actuellement sélectionnée lorsque le script est exécuté.

Exécuter la commande en tant que

Cliquez sur la flèche déroulante et sélectionnez l'option appropriée :

- Fichier de commandes - Utilisez cette option pour créer un script shell qui est exécuté dans une fenêtre de commande système ; les variables d'environnement sont accessibles par les commandes de ce script
- Processus - Utilisez cette option pour exécuter un seul programme - la commande doit spécifier le chemin d'accès au programme, ainsi que tous les arguments de la ligne de commande ; si le chemin ou les arguments contiennent des espaces, entourez le chemin d'accès de guillemets - par exemple : "c:\program files (x86)\java\bin\javac.exe"

Répertoire par défaut

La valeur par défaut est la valeur entrée pour le script de construction. Si aucune valeur n'a été définie pour le script de construction, recherchez ou saisissez le

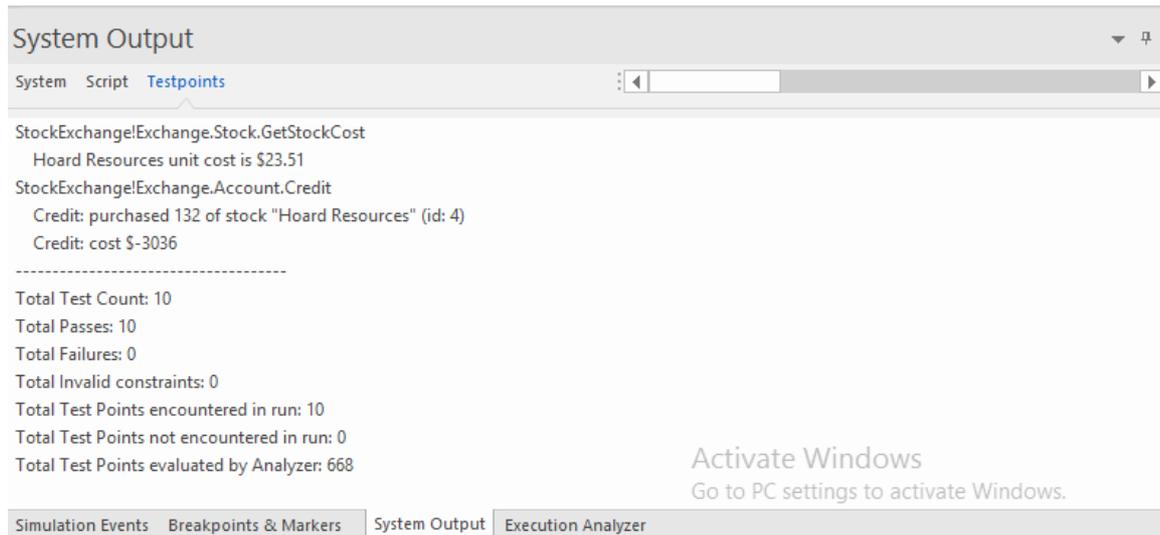
chemin du répertoire par défaut dans lequel le processus de script de nettoyage sera exécuter .

- Analyser la sortie** Lorsqu'un analyseur est sélectionné, la sortie des tests NUnit et JUnit peut être analysée, enregistrée et gérée à partir du modèle ; (Alt+2 > Tester). Sachez que la sortie n'est capturée que lorsqu'un analyseur est sélectionné.
- Hôte distant** Type l' ID du système hôte distant et son port ; par exemple, mypc01:7777.
Si vous définissez cette propriété sur #SYSTEMHOST#, le script est envoyé au Service Satellite Windows lorsqu'il est exécuté sous Windows et au Service Satellite Linux lorsqu'il est exécuté sous Wine. Les ID de service et les ports sont définis dans la section « Options privées - Services » de l'éditeur Scripts d'Analyseur .
- Construire d'abord** Sélectionnez cette option pour garantir que le Paquetage est compilé à chaque fois que vous exécutez le test.

Sortie des Testpoints

La page « Testpoints » du script Analyzer vous aide à configurer la sortie d'un exécuter Testpoint .

Par défaut, la sortie est enregistrée dans la fenêtre Sortie système, comme dans cet exemple.



Accéder

Dans la fenêtre Analyseur d'Exécution , soit :

- Recherchez et double-cliquez sur le script requis et sélectionnez la page « Test > Testpoints » ou
- Cliquez sur  dans la barre d'outils de la fenêtre, sélectionnez le Paquetage dans lequel créer un nouveau script et sélectionnez la page « Test > Testpoints ».

Ruban	Develop > Source Code > Analyseur d'Exécution > Edit Scripts d'Analyseur Exécuter > Outils > Analyseur > Vue Scripts d'Analyseur
Raccourcis Clavier	Maj+F12

Options

Option	Description
Sortir	Vous pouvez choisir entre deux options : <ul style="list-style-type: none"> • « Écran » (par défaut) - La sortie est dirigée vers l'onglet « Testpoints » de la fenêtre Sortie système • « Fichier » – La sortie est dirigée vers le fichier
Dossier	Cliquez sur le dossier à utiliser pour les fichiers log Testpoint .
Nom de fichier	Saisissez le nom à utiliser pour les fichiers log Testpoint .

Écraser	Lorsque cette option est sélectionnée, le fichier spécifié est écrasé à chaque fois qu'un exécuter Testpoint .
Numéro automatique	Lorsque cette option est sélectionnée, la sortie Testpoint est composée du nom de fichier que vous spécifiez et du numéro du Test exécuter ; chaque fois que vous effectuez un Test exécuter le numéro est incrémenté.
Sortie de trace de préfixe avec fonction	Lorsque cette option est sélectionnée, toutes les instructions de trace exécutées pendant l' exécuter Testpoint sont préfixées par l'appel de fonction en cours.

Script Débogage

Le processus de configuration de la section Déboguer d'un script Analyzer est généralement une opération ponctuelle qui doit rarement être revue. Ainsi, une fois que votre script fonctionne, vous n'aurez probablement plus à y penser. Les détails que vous fournissez ne sont pas compliqués, mais la définition d'un script donne accès à de nombreux avantages tels que :

- Débogage
- Enregistrement diagramme Séquence
- Exécution et simulation Statemachine Exécutable
- Création et enregistrement de domaines Test
- Profilage Comportementale des processus sur une variété d'exécutions

Il vous suffit de sélectionner la plateforme appropriée et de saisir quelques informations de base. Les plateformes de débogage que vous pouvez utiliser incluent :

- Java
- Protocole Java Déboguer Wire (JDWP)
- Débogueur Microsoft .NET
- Débogueur Microsoft Native Code (C++, C, VB)
- Mono
- Le Débogueur PHP
- Le Débogueur GNU (GDB)

Accéder

Ruban	Develop > Source Code > Analyseur d'Exécution > Edit Scripts d'Analyseur Exécuter > Outils > Analyseur
Raccourcis Clavier	F6

Notes

- Un script Analyzer n'est pas nécessaire pour déboguer les scripts de modèle Enterprise Architect tels que JavaScript ou VBScript

Exigences Spécifiques au Système d'Exploitation

Le débogueur Enterprise Architect peut fonctionner sur un certain nombre de plates-formes différentes. Ce tableau décrit les exigences individuelles pour le débogage sur chaque plate-forme.

Plateformes

Plate-forme	Détail
Microsoft .NET	<ul style="list-style-type: none">• Microsoft™ .NET Frameworks 4.0, 3.5 et 2.0• support des langues : C, C# , C++ , J#, VB.NET
Java	<ul style="list-style-type: none">• Kit de développement Java SE d'Oracle™ (version 5.0 minimum) (JDK 32 bits ou 64 bits) <p>L' Architecture Java Platform Débogueur (JPDA) a été introduite dans la version 5.0 de Java SE. La JPDA fournit deux protocoles de débogage : l'interface Java Virtual Machine Tools (JVMTI) et le protocole Java Débogueur Wire Protocol (JDWP).</p> <p>Le débogueur d' Enterprise Architect supporte les deux protocoles.</p>
Débogueur GNU (GDB)	<p>Enterprise Architect supporte le débogage à l'aide du GNU Débogueur , qui vous permet de déboguer vos applications sous Linux localement ou à distance.</p> <p>Nécessite la version GDB 7.0 ou supérieure.</p> <p>Le chemin du fichier de code source ne doit pas contenir d'espaces.</p>
Windows pour les applications natives	<p>Enterprise Architect supporte le débogage du code natif (C, C++ et Visual Basic) compilé avec le compilateur Microsoft™ lorsqu'un fichier PDB associé est disponible.</p>
PHP	<p>Enterprise Architect vous permet d'effectuer le débogage local et distant des scripts PHP sur les serveurs Web.</p> <p>Nécessite que le serveur Web soit configuré pour support PHP.</p> <p>Nécessite que PHP soit configuré pour support XDebug PHP (extension PHP tierce).</p>

Notes

- La facilité de débogage est disponible dans toutes les éditions d' Enterprise Architect

Systèmes d'Exploitation compatibles UAC

Le système d'exploitation Microsoft Windows 7 fournit le contrôle de compte d'utilisateur (UAC) pour gérer la sécurité des applications.

L'Analyseur d'Exécution Visuelle Enterprise Architect est compatible UAC, et les utilisateurs de systèmes compatibles UAC peuvent effectuer des opérations avec l'Analyseur d'Exécution Visuelle et facilités associées sous des comptes qui sont membres uniquement du groupe Utilisateurs.

Cependant, lors de la connexion à des processus exécutés en tant que services sur un système d'exploitation compatible UAC, il peut être nécessaire de log en tant qu'administrateur.

Connectez-vous en tant qu'administrateur

Étape	Action
1	Avant d'exécuter Enterprise Architect, cliquez-droit sur l'icône Enterprise Architect sur le bureau et sélectionnez l'option ' Exécuter en tant qu'administrateur'.

Alternativement

Modifiez ou créez un lien vers Enterprise Architect et configurez le lien pour exécuter en tant qu'administrateur.

Étape	Action
1	Cliquez-droit sur l'icône Enterprise Architect et sélectionnez l'option ' Propriétés '. La dialogue « Propriétés » Enterprise Architect s'affiche.
2	Cliquez sur le bouton Avancé. La dialogue ' Propriétés avancées' s'affiche.
3	Cochez la case « Exécuter en tant qu'administrateur ».
4	Cliquez sur le bouton OK, puis à nouveau sur la dialogue « Propriétés Enterprise Architect ».

Débogage WINE

Configurer Enterprise Architect pour déboguer sous WINE

Étape	Action
1	En ligne de commande, exécuter <code>\$ winecfg</code> .
2	Sélectionnez l'onglet « Applications ». Ajoutez l'exécutable Enterprise Architect « EA.exe » à partir du dossier d'installation Enterprise Architect . Ajoutez ensuite ces programmes à partir des sous-répertoires VEA : <ul style="list-style-type: none">• SSampler32.exe• SSampler64.exe• SSProfiler32.exe• SSProfiler64.exe
3	Sélectionnez chaque programme à tour de rôle, puis passez à l'onglet « Bibliothèques ». Assurez-vous que ces valeurs sont répertoriées avec une priorité (native, intégrée) : <ul style="list-style-type: none">• dbghelp• msxml4• msxml6
4	Copiez le code source de l'application ainsi que les exécutables sur votre bouteille. Le chemin doit être le même que la version compilée, c'est-à-dire : Si la source Windows = <code>C:\Source\SampleApp</code> , sous Crossover, ce doit être <code>C:\Source\SampleApp</code> .
5	Copiez tous les assemblages côte à côte utilisés par l'application.

Autorisations

Une installation d' Enterprise Architect contient des programmes Linux natifs qui fournissent des services de création et de débogage à Enterprise Architect sous Wine. Ces programmes doivent être vérifiés à l'aide du système de fichiers ou du shell Linux pour garantir que l'autorisation « Exécuter » est correctement définie. Les programmes se trouvent dans le sous-répertoire « VEA/x86/linux » de l'installation Enterprise Architect .

Exceptions aux violations d'accès

En raison de la manière dont WINE gère le dessin direct et l'accès aux données DIB, une option supplémentaire est fournie dans le menu déroulant de la barre d'outils de la fenêtre Déboguer pour ignorer ou traiter les exceptions de violation d'accès levées lorsque votre programme accède directement aux données DIB.

Sélectionnez cette option pour détecter les violations d'accès réelles (inattendues) ; désélectionnez-la pour ignorer les violations attendues.

Étant donné que le débogueur ne peut pas faire la distinction entre les violations attendues et inattendues, vous devrez peut-être procéder par essais et erreurs pour capturer et inspecter les véritables pannes de programme.

Notes

- Si WINE plante, les traces de retour peuvent ne pas être correctes
- Si vous utilisez MFC, n'oubliez pas de copier les assemblages côte à côte de débogage dans le répertoire C:\window\winsxs
- Pour ajouter un chemin Windows à WINE , modifiez l'entrée de registre :
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Session Manager\Environment

Java

Cette section décrit comment configurer Enterprise Architect pour le débogage des applications Java et des serveurs Web.

Configuration générale pour Java

La configuration générale pour le débogage des applications Java supporte deux options :

- Déboguer une application
- Se connecter à une application en cours d'exécution

Option 1 - Déboguer une application

Champ	Action
Débogueur	Sélectionnez Java.
x64	Cochez cette case si vous déboguez une application 64 bits. Décochez la case si vous déboguez une application 32 bits.
Mode	Sélectionnez Exécuter .
Répertoire par défaut	Ce chemin est ajouté à la propriété du chemin de classe lors de la création de la Virtual Machine Java.
Classe d'application	<p>Identifiez le nom de classe entièrement qualifié à déboguer ; la classe doit avoir une méthode déclarée avec cette signature :</p> <pre>public static void main(String ());</pre> <p>Application Class <input type="text" value="samples.Collector"/></p> <p>Command Line Arguments: <input \"param3\"="" param1\"="" param2="" param4\""="" type="text" value="\"/></p>
Arguments de la ligne de commande	<p>Spécifiez tous les paramètres à transmettre à la méthode principale de la classe d'application.</p> <p>Les paramètres contenant des espaces doivent être entourés de guillemets doubles.</p>
Options Virtual Machine Java	<p>Spécifiez les options de ligne de commande pour la création Virtual Machine .</p> <p>Vous devez également fournir un paramètre pour l'environnement d'exécution Java (JRE) comme chemin d'accès à rechercher pour le fichier jvm.dll ; il s'agit de la DLL fournie dans le cadre de l'environnement d'exécution ou du JDK de Sun Microsystems TM .</p> <p>Le paramètre JRE peut être soit :</p> <ul style="list-style-type: none"> • Un chemin local défini par un architecte d'entreprise • Un chemin de fichier absolu (sans guillemets doubles) vers le dossier d'installation du JDK Java à utiliser pour le débogage <p>Le paramètre JRE doit pointer vers le dossier d'installation du JDK Java. Une installation du JDK est nécessaire pour que le débogage réussisse. Le JRE ne doit pas pointer vers l'installation de l'environnement d'exécution Java public, si celui-ci est installé. Les variables d'environnement peuvent être utilisées lors de la spécification des options de démarrage de la machine virtuelle, telles que le chemin de classe.</p> <p>Par exemple, en utilisant :</p>

	<ul style="list-style-type: none"> • Un chemin local Enterprise Architect JAVA et un classpath de variable d'environnement : <pre>Java Virtual Machine Options: JRE=%JAVA%,-Djava.class.path=%classpath%;;</pre> • Ou un chemin absolu vers le répertoire d'installation du JDK et une variable d'environnement classpath : <pre>Java Virtual Machine Options: JRE=C:\Program Files (x86)\Java\jdk1.7.0,-Djava.class.path=%classpath%;;</pre> <p>Dans ces deux exemples, le débogueur va créer une machine virtuelle en utilisant le JDK situé à la valeur du paramètre JRE.</p> <p>Si aucun chemin de classe n'est spécifié, le débogueur crée toujours la machine virtuelle avec une propriété de chemin de classe égale à tout chemin contenu dans la variable d'environnement plus le chemin entré dans le répertoire de travail par défaut de ce script.</p> <p>Si les fichiers source et les fichiers .class sont situés sous des arborescences de répertoires différentes, la propriété classpath DOIT inclure les chemins racine vers la source et les chemins racine vers les fichiers de classe binaires.</p>
--	---

Option 2 – Se connecter à Virtual Machine

Il y a très peu de choses à spécifier lors de la connexion à une VM ; cependant, l'agent de débogage Sparx Systems doit être chargé sur la VM.

Champ	Action
Débogueur	Sélectionnez Java
Mode	Sélectionnez Attacher à Virtual Machine

Techniques avancées

En plus des techniques de débogage Java standard, vous pouvez :

- [Attach to Virtual Machine](#)
- [Internet Browser Java Applets](#)

Se connecter à Virtual Machine

Vous pouvez déboguer une application Java en vous connectant à un processus qui héberge une Virtual Machine Java ; vous souhaitez peut-être le faire pour vous connecter à un serveur Web tel que Tomcat ou JBOSS.

L'interface Java Virtual Machine Tools de Sun Microsystems est l'API utilisée par Enterprise Architect ; elle permet de spécifier un agent de débogage lors de la création de la JVM.

Pour déboguer une JVM en cours d'exécution à partir d' Enterprise Architect , l'agent de débogage de Sparx Systems doit avoir été spécifié comme option de démarrage de la JVM lors de son démarrage ; la manière dont cela est réalisé pour des produits tels que Tomcat et JBOSS doit être fournie par la documentation de ce produit.

Pour java.exe, l'option de ligne de commande pour charger l'agent de débogage Enterprise Architect pourrait être (selon votre environnement) :

- -agentpath:"c:\program files\sparx systems\ea\VEA\x86\SSJavaProfiler32"
- -agentpath:"c:\program files (x86)\sparx systems\ea\VEA\x86\SSJavaProfiler32"
- -agentpath:"c:\program files (x86)\sparx systems\ea\VEA\x64\SSJavaProfiler64"

L'option appropriée dépendra de votre système d'exploitation et du fait que vous travaillez sur une application 32 bits ou 64 bits.

Alternativement, si vous ajoutez le répertoire VEA approprié à votre variable d'environnement PATH, vous pouvez choisir d'utiliser :

- -agentlib:SSJavaProfiler32
- -agentlib:SSJavaProfiler64

Il n'est pas nécessaire de configurer un script Analyzer lorsque vous vous connectez à une Virtual Machine ; vous pouvez simplement utiliser le bouton Attacher sur l'une des barres d'outils Analyzer.

Si vous configurez un script d'analyse, seules deux choses doivent être sélectionnées :

- Sélectionnez « Java » comme plate-forme de débogage
- Choisissez l'option « Attacher à Virtual Machine »

Applet Java Navigateur Internet

Cette rubrique décrit les exigences de configuration et la procédure de débogage des applets Java exécutées dans un navigateur à partir d' Enterprise Architect .

Connectez-vous au processus du navigateur hébergeant la Virtual Machine Java (JVM) depuis Enterprise Architect

Étape	Action
1	Assurez-vous que les binaires du code de l'applet à déboguer ont été créés avec les informations de débogage.
2	Configurez la JVM à l'aide du panneau de configuration Java.
3	Dans le panneau « Paramètres d'exécution de l'applet Java », cliquez sur le bouton Vue .
4	Sur la version installée à utiliser, incluez l'une de ces options dans le champ « Paramètres d'exécution », en fonction de votre environnement et selon que vous travaillez sur une application 32 bits ou une application 64 bits : -agentpath:"c:\program files\sparx systems\ea\VEA\x86\SSJavaProfiler32" -agentpath:"c:\program files (x86)\sparx systems\ea\VEA\x86\SSJavaProfiler32" -agentpath:"c:\program files (x86)\sparx systems\ea\VEA\x64\SSJavaProfiler64"
5	Dans ce champ, ajoutez les chemins de classe requis. Au moins un de ces chemins doit inclure le chemin racine des fichiers sources à utiliser dans le débogage.
6	Définir des points d'arrêt.
7	Lancez le navigateur.
8	Attachez-vous au processus du navigateur depuis Enterprise Architect .

Travailler avec Serveurs Web Java

Si vous déboguez des serveurs Web Java tels que JBOSS et Apache Tomcat (configuration du serveur et configuration du service Windows) dans Enterprise Architect , appliquez ces exigences et procédures de configuration.

Note : Les fonctionnalités de débogage et d'enregistrement de l' Analyseur d'Exécution Visuelle ne sont pas supportées pour la plateforme serveur Java 'Weblogic' d'Oracle.

Se connecter au processus hébergeant la Virtual Machine Java depuis Enterprise Architect

Étape	Action
1	Créez des binaires pour le code du serveur Web à déboguer, avec des informations de débogage.
2	Lancez le serveur avec l'option « Démarrage Virtual Machine », décrite dans <i>Configuration du serveur</i> .
3	Importez le code source dans le Modèle Enterprise Architect ou synchronisez le code existant.
4	Définir des points d'arrêt.
5	Lancer le client.
6	Attachez-vous au processus depuis Enterprise Architect .

Configuration du serveur

La configuration nécessaire pour que les serveurs Web puissent interagir avec Enterprise Architect doit répondre à ces deux points essentiels :

- Toute machine virtuelle à déboguer, créer ou héberger par le serveur doit avoir l'option de ligne de commande Sparx Systems Agent spécifiée ou dans l'option de démarrage de la machine virtuelle (c'est-à-dire :
-agentlib:SSJavaProfiler32 ou -agentlib:SSJavaProfiler64)
- Le CLASSPATH, quelle que soit la manière dont il est transmis à la machine virtuelle, doit spécifier le chemin racine vers les fichiers sources Paquetage

Le débogueur Enterprise Architect utilise la propriété java.class.path dans la machine virtuelle en cours de débogage pour localiser le fichier source correspondant à un point d'arrêt se produisant dans une classe pendant l'exécution ; par exemple, une classe à déboguer est appelée :

abc

Ceci est situé dans le répertoire physique :

C:\source\ab

Ainsi, pour que le débogage réussisse, le CLASSPATH doit contenir le chemin racine :

c:\source

Configuration du script d'analyse

En utilisant l'onglet « Débuguer » de la dialogue « Build Script », créez un script pour le code que vous avez importé et :

- Sélectionnez le bouton radio « Joindre au processus » et, dans le champ situé en dessous, saisissez « joindre »
- Dans le champ « Utiliser Débugueur », cliquez sur la flèche déroulante et sélectionnez « Java »

Tous les autres champs ne sont pas importants ; le champ « Répertoire » est normalement utilisé en l'absence de toute propriété de chemin de classe.

Exécuter le Débugueur

Les points d'arrêt peuvent afficher un point d'interrogation. Dans ce cas, la classe n'a peut-être pas encore été chargée par la machine virtuelle. Si le point d'interrogation persiste même après que vous êtes sûr que la classe contenant le point d'arrêt a été chargée, alors :

- Les binaires exécutés par le serveur ne sont pas basés sur le code source
- Le débogueur ne peut pas réconcilier le point d'arrêt avec un fichier source (vérifiez les chemins de classe), ou
- La JVM n'a pas chargé l'agent Sparx Systems

Étape	Action
1	Exécuter le serveur et vérifiez que le processus serveur a chargé l'agent Sparx Systems : DLL SSJavaProfiler32.DLL ou SSJavaProfiler64 Utilisez « Process Explorer » ou des outils similaires pour prouver que le processus serveur a chargé l'agent.
2	Dans Enterprise Architect , ouvrez le code source et définissez des points d'arrêt.
3	Cliquez sur le bouton Exécuter Débuguer dans Enterprise Architect . La dialogue « Attacher au processus » s'affiche.
4	Sélectionnez le processus serveur hébergeant l'application.
5	Cliquez sur le bouton OK . Un message de confirmation s'affiche dans la fenêtre Débuguer , indiquant que le processus a été attaché.

Serveur JBOSS

Dans cet exemple JBoss, pour une application 32 bits, le code source d'un servlet simple se trouve dans le répertoire :

C:\Benchmark\Java\JBOSS\Inventaire

Les binaires exécutés par JBOSS se trouvent dans le fichier JAW.EAR à cet emplacement :

C:\JBOSS\03b-dao\build\distribution

Le débogueur Enterprise Architect doit être capable de localiser les fichiers sources pendant le débogage ; pour ce faire, il utilise également le CLASSPATH, en recherchant dans n'importe quel chemin répertorié un fichier source JAVA correspondant, de sorte que le CLASSPATH doit inclure un chemin vers la racine du Paquetage pour Enterprise Architect trouve la source pendant le débogage.

Ceci est un extrait du fichier de commandes qui exécute le serveur JBOSS ; la classe à déboguer se trouve à :

com/inventaire/dto/carDTO

Par conséquent, la racine de ce chemin est incluse dans JBOSS_CLASSPATH.

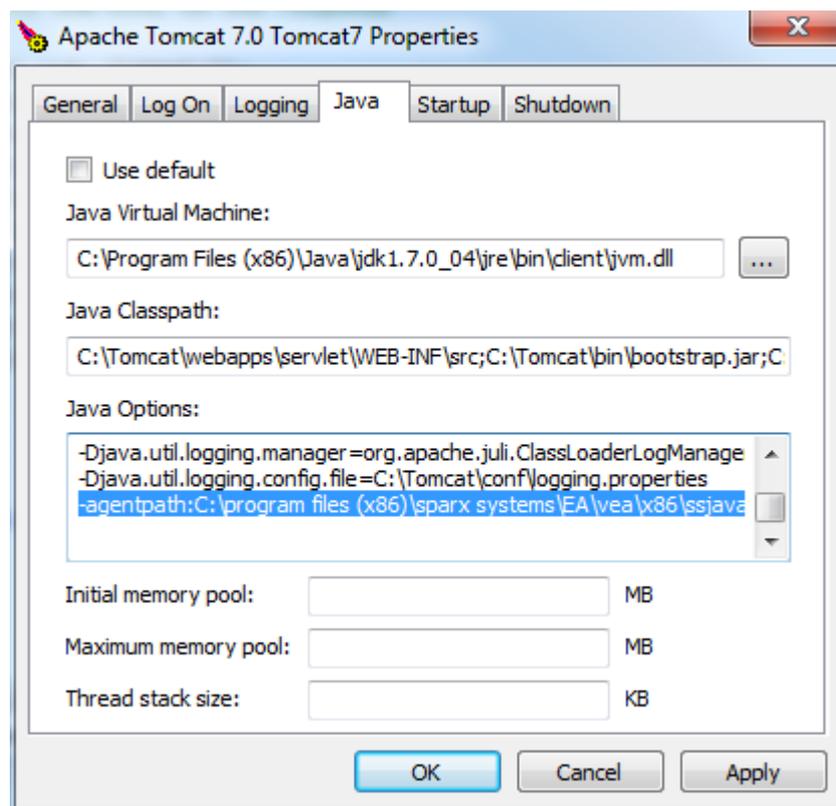
Exemple de code

RUN.BAT

```
set SOURCE=C:\Benchmark\Java\JBOSS\Inventory
set JAVAC_JAR=%JAVA_HOME%\lib\tools.jar
if "%JBOSS_CLASSPATH%" == ""
(
set JBOSS_CLASSPATH=%SOURCE%;%JAVAC_JAR%;%RUNJAR%;
)
else
(
set JBOSS_CLASSPATH=%SOURCE%;%JBOSS_CLASSPATH%;%JAVAC_JAR%;%RUNJAR%;
)
set JAVA_OPTS=%JAVA_OPTS% -agentpath:"c:\program files\sparx systems\vea\x86\ssjavaprofiler32"
```

Serveur Apache Tomcat

Le serveur Apache Tomcat peut être configuré pour le débogage à l'aide du débogueur Java dans Enterprise Architect . Cet exemple montre la dialogue de configuration d'Apache Tomcat 7.0 sur un PC exécutant Windows 7 .



Ces trois points sont importants :

- La « Virtual Machine Java » spécifie le temps d'exécution à partir d'une installation du JDK Java
- Le chemin source de tout servlet à déboguer est ajouté au Classpath Java ; dans ce cas, nous ajoutons le chemin vers le servlet Tomcat :
c:\tomcat\webapps\servlet\WEB-INF\src
- Les « Options Java » incluent le chemin d'accès à l'agent de débogage Sparx Systems :
-agentpath:c:\program files (x86)\sparx systems\vea\x86\ssjavaprofiler32

Apache Tomcat Windows Service

Configuration

Pour les utilisateurs exécutant Apache Tomcat en tant que service Windows TM, il est important de configurer le service pour permettre l'interaction avec le bureau ; le non-respect de cette consigne entraîne l'échec du débogage dans Enterprise Architect .

Log on as:

Local System account

Allow service to interact with desktop

Cochez la case « Autoriser le service à interagir avec le bureau ».

.NET

Cette section décrit comment configurer Enterprise Architect pour le débogage des applications .NET . Elle comprend :

- [General Setup for .NET](#)
- [Debugging an Unmanaged Application](#)
- [Debug COM Interop](#)
- [Debug ASP .NET](#)

Configuration générale pour .NET

Il s'agit de la configuration générale pour le débogage des applications Microsoft .NET . Vous avez deux options pour le débogage :

- Déboguer une application
- Se connecter à une application en cours d'exécution

Option 1 – Déboguer une application

Champ	Action
Débogueur	Sélectionnez Microsoft .NET comme plate-forme de débogage.
x64	Cochez cette case si vous déboguez une application 64 bits. Décochez la case si vous déboguez une application 32 bits.
Mode	Sélectionnez le bouton radio Exécuter .
Répertoire par défaut	Ceci est défini comme répertoire par défaut pour le processus en cours de débogage.
Chemin d'application	Sélectionnez et entrez le chemin complet ou relatif vers l'exécutable de l'application. <ul style="list-style-type: none"> • Si le chemin contient des espaces, spécifiez le chemin complet ; n'utilisez pas de chemin relatif • Si le chemin contient des espaces, le chemin doit être entouré de guillemets
Arguments de la ligne de commande	Paramètres à passer à l'application au démarrage.
Afficher la console	Créer une fenêtre de console pour le débogueur ; ne s'applique pas à la connexion à un processus.
Chemins de recherche de symboles	Spécifiez tous les chemins supplémentaires pour localiser les symboles de débogage pour le débogueur ; séparez les chemins par un point-virgule.

Option 2 – Se connecter à une application en cours d'exécution

Champ	Action
Débogueur	Sélectionnez Microsoft .NET comme plate-forme de débogage.
x64	Cochez cette case si vous déboguez une application 64 bits. Décochez la case si vous déboguez une application 32 bits.

Mode	Sélectionnez le bouton radio Attacher au processus.
------	---

Débogage d'une application non gérée

Si vous déboguez du code managé à l'aide d'une application non gérée, le débogueur risque de ne pas détecter la version correcte du Common Language Runtime (CLR) à charger.

Vous devez spécifier un fichier de configuration si vous n'en avez pas déjà un pour l'application de débogage spécifiée dans la commande Déboguer de votre script.

Le fichier de configuration doit résider dans le même répertoire que votre application et prendre le format :

```
name.exe.config
```

où « nom » est le nom de votre application.

La version du CLR que vous spécifiez doit correspondre à la version chargée par le code managé invoqué par le débogueur.

Dans l'exemple de code qui suit, « clr_version » représente la version du CLR ciblée par votre plug-in ou votre code COM.

Exemple de fichier de configuration

```
<configuration>
  <startup>
    <requiredRuntime version="clr_version"/>
  </startup>
</configuration>
```

Déboguer COM Interop

Enterprise Architect vous permet de déboguer le code géré .NET exécuté à l'aide de COM sur un serveur local ou en cours de processus.

Cette fonctionnalité est utile pour déboguer les plug-ins et les composants ActiveX.

Déboguer le code managé .NET exécuté à l'aide de COM

Étape	Action
1	Créer un Paquetage dans Enterprise Architect et importez le code à déboguer.
2	Assurez-vous que le composant COM est créé avec des informations de débogage.
3	Créer un script pour le Paquetage .
4	Dans la page « Déboguer Plateforme », vous pouvez choisir de vous attacher à un processus non géré ou de spécifier le chemin d'accès à une application non gérée pour appeler votre code géré.
5	Ajoutez des points d'arrêt dans le code source pour déboguer.

Attacher à un processus non géré

Si vous utilisez :

- Un serveur COM en cours de processus, attaché au processus client
- Un serveur COM local, attaché au processus serveur

Cliquez sur le bouton Exécuter de la fenêtre Déboguer (ou appuyez sur F6) pour afficher une liste de processus parmi lesquels vous pouvez choisir.

Notes

- Le détachement d'un processus d'interopérabilité COM que vous avez débogué met fin au processus ; il s'agit d'un problème connu pour Microsoft .NET Framework, et des informations à ce sujet sont disponibles sur de nombreux blogs MSDN .NET

Déboguer ASP .NET

Le débogage des services Web tels que ASP nécessite que le débogueur Enterprise Architect soit capable de se connecter à un service en cours d'exécution.

Commencez par vous assurer que le répertoire contenant le projet de service ASP .NET a été importé dans Enterprise Architect et, si nécessaire, le dossier Web contenant les pages Web du client.

Si le répertoire de votre projet Web se trouve sous le répertoire d'hébergement du site Web, vous pouvez importer à partir de la racine et inclure à la fois le code ASP et les pages Web.

Il est nécessaire de lancer d'abord le client, car le processus de service ASP .NET n'est peut-être pas déjà en cours d'exécution ; chargez le client à l'aide de votre navigateur : cela garantit que le serveur Web est en cours d'exécution.

Dans la configuration de débogage, vous devez ensuite sélectionner le bouton radio « Attacher ». Lorsque ce choix est sélectionné, le débogueur vous prompt à chaque fois de déboguer le processus.

Cliquez sur le bouton Exécuter de la fenêtre Déboguer pour démarrer le débogueur ; la dialogue « Attacher au processus » s'affiche.

Le nom du processus varie selon les systèmes d'exploitation Microsoft, comme expliqué dans le *SDK ASP .NET* ; par exemple, sur Windows XP, le nom du processus ressemble à aspnet_wp.exe, bien que le nom puisse refléter la version du framework .NET qu'il prend en charge.

Plusieurs processus ASP.NET peuvent s'exécuter sous XP ; vous devez vous assurer que vous vous connectez à la bonne version, qui sera celle hébergeant la version de .NET Framework sur laquelle votre application s'exécute ; vérifiez le fichier web.config de votre service Web pour vérifier la version de .NET Framework à laquelle il est lié.

Le bouton Arrêter de la fenêtre Déboguer doit être activé et tous les points d'arrêt doivent être rouges, indiquant qu'ils ont été liés.

Vous pouvez définir des points d'arrêt à tout moment dans le code du serveur Web. Vous pouvez également définir des points d'arrêt dans les pages Web ASP si vous les avez importées.

Notes

Certains points d'arrêt n'ont peut-être pas été liés avec succès, mais si aucun n'est lié (indiqué par un rouge foncé avec des points d'interrogation), quelque chose n'est plus synchronisé ; essayez de reconstruire et de réimporter le code source

Le Débogueur Mono

Mono est une plateforme logicielle sponsorisée par la .NET Foundation pour faciliter le développement multiplateforme. Elle est populaire auprès des développeurs de jeux pour ses fonctionnalités riches en termes de jeu, basées sur des API et de portabilité.

Enterprise Architect apporte support à la communauté Mono en fournissant un environnement moderne pour modélisation et le développement de logiciels. Les projets existants peuvent être importés, créés et débogués de manière native sous Linux et Windows .

Aperçu

Le débogage sous Mono implique la coopération de trois processus. Le runtime Mono gère l'application et communique à l'aide d'un protocole socket avec le Débogueur Enterprise Architect , qui à son tour communique avec Enterprise Architect en tant que front-end. Lorsque vous lancez Mono, vous devez lui demander de supporter le débogage, ce que vous faites à l'aide d'une directive de ligne de commande dans laquelle vous nommez l'hôte et le numéro de port sur lesquels Mono doit écouter. L'hôte peut être omis, auquel cas Mono acceptera les connexions à partir de n'importe quelle adresse IP. L'hôte peut avoir la valeur 'localhost' pour restreindre les connexions à la même machine. Le numéro de port est un numéro de votre choix.

L'hôte et le numéro de port sont des informations importantes, car ils sont utilisés lors de la configuration du script Analyzer.

Exigences pour Windows

- Enterprise Architect (version 14 minimum)
- Mono pour Windows (version 5.4 minimum)

Exigences pour Linux

- Enterprise Architect (version 14 minimum)
- Mono pour Linux (version 5.4 minimum)
- Wine pour Linux

La page hôte d'exécution

Cette page est facultative et n'est utile que si Mono et Enterprise Architect s'exécutent sur la même machine. Elle permet d' exécuter d'abord Mono avec les directives de débogage requises, avant de démarrer le débogueur Enterprise Architect . Une fois que le débogueur s'est connecté, il reprend l'exécution de Mono, qui a été démarrée comme suspendue. Si l'application s'exécute sur une machine différente de l' Enterprise Architect que vous utilisez, vous devez effacer cette section.

Configuration de débogage Linux

Configuration Débogueur

Cette section décrit la section Débogueur d'un script Analyzer en ce qui concerne le débogage de Mono sous Linux. Les champs qui ne sont pas répertoriés ici ne sont pas obligatoires.

Débogueur	Sélectionnez « Mono ».
Répertoire par défaut	Il s'agit du chemin Linux natif entièrement qualifié où se trouve l'application au format Unix.
Connexion	<ul style="list-style-type: none"> port : le port de débogage hôte : le nom ou l'adresse IP de la machine sur laquelle Mono s'exécute (« localhost » si la machine est la même) localpath : le chemin racine du code source au format Windows ; il s'agit du chemin vers les fichiers sources que vous utilisez pour définir des points d'arrêt dans l'éditeur de code d' Enterprise Architect remotepath : le chemin racine du code source au format Unix, c'est-à-dire le chemin vers les fichiers sources utilisés pour construire le programme sous Linux <p>Ces chemins sont renvoyés lors des événements de débogage, puis mappés au chemin local, afin qu'Enterprise Architect puisse afficher le fichier source lors d'un point d'arrêt ou d'une étape - les deux paramètres peuvent spécifier la même racine de fichier source physique, mais doivent utiliser le format Windows ou Unix pour chaque champ</p> <ul style="list-style-type: none"> shutdown : (true ou false) ; lorsque true la VM est terminée lorsque le Débogueur est arrêté timeout : le délai d'attente en millisecondes pour les appels de socket sortie : le chemin Wine / Windows du fichier log dans lequel écrire logging : (true ou false) ; lorsque la valeur est true, des messages supplémentaires sont enregistrés dans la fenêtre Débogueur et les messages de socket sont enregistrés dans le fichier de sortie spécifié

Démarrage automatique de Mono

Vous pouvez configurer Enterprise Architect pour qu'il démarre Mono à votre place lorsque vous démarrez le débogueur. Pour ce faire, configurez la page « Hôte d'exécution » de votre script Analyzer. Le format des commandes est décrit ici :

chemin d'accès au programme cd

```
/usr/bin/mono --debug --debugger-agent=transport=dt_socket,address= hôte:port ,serveur=y,suspend=y programme
```

où:

- *path-to-program* est le chemin du répertoire où se trouve le programme

- *l'hôte* est l'un de ceux-ci :

- hôte local

- une adresse IP
 - un nom de machine en réseau
- *port* est le port pour le socket
- *programme* est le nom de l'application (par exemple MonoProgram.exe)

Démarrage manuel de Mono à l'aide de la ligne de commande

Vous pouvez démarrer Mono manuellement à partir d'une console. Localisez le programme dans votre explorateur de fichiers, puis ouvrez une console à cet emplacement. Le format de la ligne de commande est décrit ici :

```
/usr/bin/mono --debug --debugger-agent=transport=dt_socket,address= hôte:port ,serveur=y,suspend=y programme
```

où *l'hôte* est l'un de ceux-ci :

- hôte local
- une adresse IP
- un nom de machine en réseau

port est le port du socket et *programme* est le nom de l'application (par exemple, MonoProgram.exe).

Fenêtres de configuration de Windows

Configuration Débogueur

Cette section décrit la section Débuguer d'un script Analyzer en ce qui concerne le débogage de Mono sous Windows . Les champs qui ne sont pas répertoriés ici ne sont pas obligatoires.

Champ	Description
Débogueur	Sélectionnez « Mono ».
x64	Sélectionnez si le programme à déboguer est un exécutable 64 bits.
Exécuter ou Attacher	Choisissez « Exécuter » pour nommer le programme à lancer. Choisissez « Attacher » si vous souhaitez toujours vous attacher à un processus en cours d'exécution.
Répertoire par défaut	Le répertoire par défaut que le programme prendra lors de son exécution.
Chemin d'application	Le chemin complet de l'application Mono.
Arguments de la ligne de commande	Tous les paramètres à transmettre au programme. Si les paramètres contiennent des espaces, entourez-les de guillemets doubles (")

Démarrage automatique de Mono

Vous pouvez configurer Enterprise Architect pour qu'il démarre Mono à votre place lorsque vous démarrez le débogueur. Pour ce faire, configurez la page « Hôte d'exécution » de votre script Analyzer. Le format des commandes est décrit ici :

chemin d'accès au programme cd

mono --debug --debugger-agent=transport=dt_socket,address=*hôte:port* ,serveur=y,suspend=y *programme*

où:

- *path-to-program* est le chemin du répertoire où se trouve le programme

- *l'hôte* est l'un de ceux-ci :

- hôte local
- une adresse IP
- un nom de machine en réseau

- *port* est le port pour le socket

- *programme* est le nom de l'application (par exemple MonoProgram.exe)

Démarrage manuel de Mono à l'aide de la ligne de commande

Vous pouvez démarrer Mono manuellement à partir d'une console. Localisez le programme dans votre explorateur de fichiers, puis ouvrez une console à cet emplacement. Le format de la ligne de commande est décrit ici :

mono --debug --debugger-agent=transport=dt_socket,address=*hôte:port* ,serveur=y,suspend=y *programme*

où *l'hôte* est l'un de ceux-ci :

- hôte local
- une adresse IP
- un nom de machine en réseau

port est le port du socket et *programme* est le nom de l'application (par exemple, MonoProgram.exe).

Le Débogueur PHP

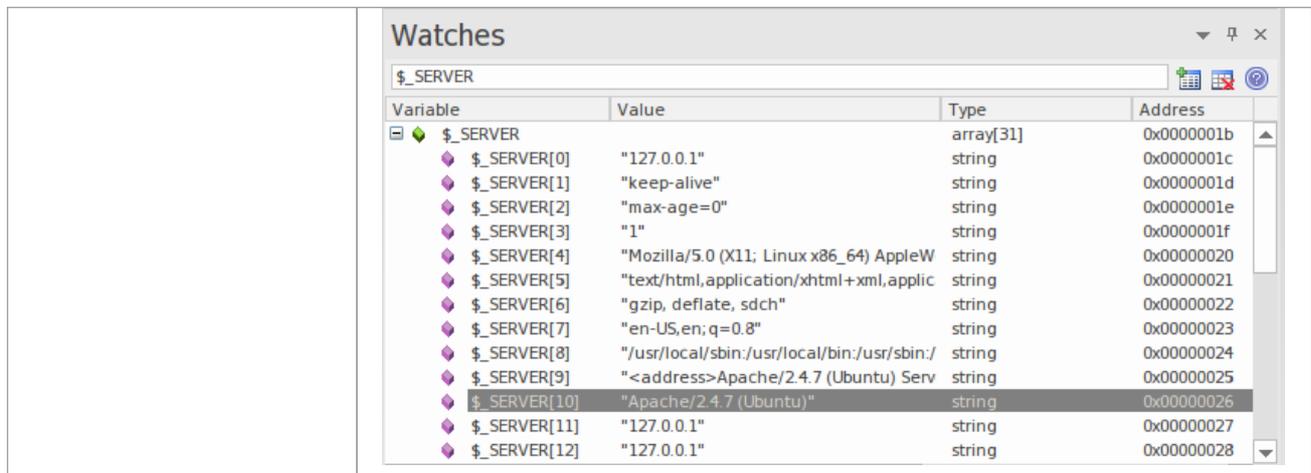
Le Débogueur PHP Enterprise Architect vous permet de déboguer les scripts PHP.exe. Cette section décrit la configuration de base et les différents scénarios de débogage fréquemment rencontrés. Ces scénarios concernent le mappage des chemins de fichiers, qui est essentiel au succès d'une session de débogage à distance.

- Configuration du script
- Machine Windows locale (serveur Apache)
- Machine Windows locale (PHP.exe)
- Machine Linux distante (serveur Apache)
- Machine Linux distante (PHP.exe)

Configuration et scénarios

Scénario	Détails
Configuration du script	<p>Un script d'analyse est une exigence de base pour le débogage dans Enterprise Architect ; vous créez un script en utilisant la barre d'outils de l' Analyseur d'Exécution .</p> <p>Sélectionnez PHP.XDebug comme plate-forme de débogage ; lorsque vous sélectionnez cette plate-forme, la page de propriétés affiche ces paramètres de connexion :</p> <ul style="list-style-type: none"> • hôte - localhost - L'adaptateur sur lequel Enterprise Architect écoute les connexions entrantes de PHP • localpath - %LOCAL% - Spécifie le chemin d'accès au fichier local à mapper sur un chemin d'accès au fichier distant ; il s'agit d'un paramètre de débogage à distance - pour le débogage local, effacez la valeur , la valeur est un espace réservé et vous devez la modifier pour l'adapter à votre scénario particulier • remotepath - %REMOTE% - Spécifie le chemin d'accès au fichier distant auquel un chemin d'accès au fichier local doit être mappé ; il s'agit d'un paramètre de débogage à distance - pour le débogage local, effacez la valeur , la valeur est un espace réservé et vous devez la modifier pour l'adapter à votre scénario particulier • journalisation - Entrez true ou false pour activer la journalisation des communications depuis le serveur XDebug • sortie - nomme le chemin du fichier sur la machine distante à utiliser avec l'option de journalisation ; ce fichier sera toujours écrasé
Serveur Apache sur machine locale	<p>Dans cette situation, considérez cette configuration :</p> <ul style="list-style-type: none"> • S/E: Windows 7 • Nom de l'ordinateur réseau : MyPC • Partage réseau MyShare mappé sur c:\myshare • Les fichiers sources dans Enterprise Architect ont été importés depuis c:\myshare\apache\myapp\scripts • La racine du document Apache est définie sur //MyPC/MyShare/apache <p>Dans ce scénario, un script d'analyse pour les paramètres de connexion peut être configuré comme suit :</p> <ul style="list-style-type: none"> • hôte : localhost • port: 9000

	<ul style="list-style-type: none"> • chemin local : c:\myshare\apache\ • chemin distant : MonPC/MonPartage/apache/
Machine locale PHP.EXE	<p>Dans ce scénario, un script d'analyse pour les paramètres de connexion peut être configuré comme indiqué, car les chemins de fichiers correspondent toujours au même chemin physique :</p> <ul style="list-style-type: none"> • hôte : localhost • port: 9000 • chemin local : • chemin distant :
Serveur Apache sur machine Linux distante	<p>Dans cette situation, considérez cette configuration :</p> <p>Machine locale :</p> <ul style="list-style-type: none"> • S/E: Windows 7 • Les fichiers sources dans Enterprise Architect ont été importés depuis c:\myshare\apache\myapp\scripts <p>Machine distante :</p> <ul style="list-style-type: none"> • S/E: Linux • La racine du document Apache est définie sur home/apache/htdocs • Les fichiers sources dans Apache se trouvent dans home/apache/htdocs/myapp/scripts <p>Dans ce scénario, un script d'analyse pour les paramètres de connexion peut être configuré comme suit :</p> <ul style="list-style-type: none"> • hôte : localhost • port: 9000 • chemin local : c:\myshare\apache\ • chemin distant : home/apache/htdocs/
Machine Linux distante PHP.exe	<p>Dans cette situation, considérez cette configuration :</p> <ul style="list-style-type: none"> • Machine locale • S/E: Windows 7 • Les fichiers sources dans Enterprise Architect ont été importés depuis c:\myshare\apache\myapp\scripts • Machine à distance • S/E: Linux • Fichiers sources dans Apache situés dans home/myapp/scripts <p>Dans ce scénario, un script d'analyse pour les paramètres de connexion peut être configuré comme suit :</p> <ul style="list-style-type: none"> • hôte : localhost • port: 9000 • chemin local : c:\myshare\apache\ • chemin distant : accueil/
Variables globales PHP	<p>Lorsque vous êtes à un point d'arrêt, vous pouvez examiner les valeurs des variables globales PHP en utilisant la fenêtre Analyzer Observateurs . Pour lister toutes les variables globales, tapez soit « globals » soit « superglobals » dans le champ. Pour afficher un élément individuel, entrez son nom. Cette image montre la valeur de la variable d'environnement PHP \$_SERVER affichée.</p>



The screenshot shows a debugger's 'Watches' window. The title bar reads 'Watches' and the search bar contains '\$_SERVER'. Below the search bar is a table with four columns: 'Variable', 'Value', 'Type', and 'Address'. The table lists the elements of the \$_SERVER array, with the 10th element (index 9) highlighted in grey.

Variable	Value	Type	Address
\$_SERVER		array[31]	0x0000001b
\$_SERVER[0]	"127.0.0.1"	string	0x0000001c
\$_SERVER[1]	"keep-alive"	string	0x0000001d
\$_SERVER[2]	"max-age=0"	string	0x0000001e
\$_SERVER[3]	"1"	string	0x0000001f
\$_SERVER[4]	"Mozilla/5.0 (X11; Linux x86_64) AppleW"	string	0x00000020
\$_SERVER[5]	"text/html,application/xhtml+xml,applic"	string	0x00000021
\$_SERVER[6]	"gzip, deflate, sdch"	string	0x00000022
\$_SERVER[7]	"en-US,en;q=0.8"	string	0x00000023
\$_SERVER[8]	"/usr/local/sbin:/usr/local/bin:/usr/sbin/"	string	0x00000024
\$_SERVER[9]	"<address>Apache/2.4.7 (Ubuntu) Serv"	string	0x00000025
\$_SERVER[10]	"Apache/2.4.7 (Ubuntu)"	string	0x00000026
\$_SERVER[11]	"127.0.0.1"	string	0x00000027
\$_SERVER[12]	"127.0.0.1"	string	0x00000028

Débogueur PHP - Exigences système

Cette rubrique identifie la configuration système requise et les systèmes d'exploitation pour le débogueur PHP Enterprise Architect .

Exigences du système :

- Enterprise Architect version 9
- Version PHP 5.3 ou supérieure
- Extension PHP Zend XDebug 2.1 ou supérieur
- Pour les serveurs Web tels qu'Apache, une version de serveur supporte la version PHP

Systèmes d'Exploitation supportés :

- Client (Enterprise Architect)
- Microsoft Windows XP et supérieur
- Linux exécutant Crossover Office
- Serveur (PHP)
- Microsoft Windows XP et supérieur
- Linux

Liste de contrôle Débogueur PHP

Cette rubrique fournit un guide de dépannage pour le débogage des scripts PHP dans Enterprise Architect .

Points de contrôle

Point de contrôle	Détails
Exigences du système	<ul style="list-style-type: none"> • Serveur Web HTTP Apache version 2.2 • Version PHP 5.3 ou supérieure • Version 2.1.1 de XDebug
Enterprise Architect	<ul style="list-style-type: none"> • Le modèle dispose d'un script d'analyse configuré pour utiliser la plateforme PHP XDebug • Le code source PHP a été importé dans le modèle (pour l'enregistrement et les points de test) • Lorsque la plateforme PHP XDebug est sélectionnée dans la dialogue « Script d'analyse », les paramètres d'exécution par défaut sont répertoriés dans le champ « Connexion » : chemin local:%LOCAL% chemin distant : %REMOTE% Définissez des chemins locaux pour ces variables par défaut ou modifiez le script pour fournir des chemins réels. Par exemple : source locale, source distante chemin local : c:\exemples de code\vea\php\exemple chemin distant : serveur Web/exemple • « serveur Web » est un réseau ou un partage local • 'sample' est un dossier sous share
PHP	<p>Afin de déboguer les scripts PHP dans Enterprise Architect , il est nécessaire que PHP soit configuré correctement pour charger l'extension XDebug.</p> <p>Des paramètres similaires à ceux-ci doivent être utilisés (pour XDebug version 3 ou supérieure) :</p> <ul style="list-style-type: none"> • [xdebug] • zend_extension=xdebug.so • xdebug.mode=débogage • xdebug.mode=débogage • xdebug.start_with_request=oui • xdebug.client_host=localhost • xdebug.client_port=9003 <p>Pour les versions Xdebug inférieures à 3, utilisez les anciens paramètres tels que :</p> <ul style="list-style-type: none"> • [xdebug] • xdebug.extended_info=1 • xdebug.idekey=ea • xdebug.remote_enable=1 • xdebug.remote_handler=dbgp

	<ul style="list-style-type: none"> • xdebug.remote_autostart=1 • xdebug.remote_host=XXXX • xdebug.remote_port=9000 • xdebug.show_local_vars=1 <p>L'adresse IP XXXX fait référence à l'hôte spécifié dans le script d'analyse du modèle et doit correspondre.</p> <p>L'adresse IP est l'adresse à laquelle XDebug se connecte et la même adresse que l'agent PHP Enterprise Architect écoute.</p>
Apache	<p>Pour le débogage à l'aide d'Apache, ces lignes doivent être présentes dans le fichier de configuration Apache, httpd.conf :</p> <p>Charger le module php5_module "php_home/php5apache2_2.dll"</p> <p>Application AddHandler/x-httpd-php .php</p> <p>PHPIniDir "php_home"</p> <p>La valeur "php_home" est le chemin d'installation de PHP (le chemin où existent php.ini et apache dll).</p>
Dépannage	<p>Pour éviter les dépassements de délai de PHP et d'Apache pendant une session de débogage, ces paramètres peuvent nécessiter une modification.</p> <p>Les paramètres ont été utilisés lors du développement de l'agent de débogage PHP dans Enterprise Architect .</p>
PHP	<p>Fichier: php.ini</p> <p>; Enterprise Architect évite les dépassements de délai PHP lors du débogage des extensions PHP</p> <p>max_execution_time = 0</p> <p>; Enterprise Architect évite les dépassements de délai du serveur Web lors du débogage des extensions PHP</p> <p>max_input_time = -1</p> <p>; Enterprise Architect enregistre les erreurs</p> <p>display_errors = Activé</p> <p>; Enterprise Architect affiche des erreurs de démarrage</p> <p>display_startup_errors = Activé</p>
Apache	<p>Fichier : httpd.conf</p> <p>; Enterprise Architect évite les dépassements de délai lors du débogage des extensions PHP</p> <p>Délai d'attente 60000</p>

Le Débogueur GNU (GDB)

Pour déboguer vos applications, vous pouvez utiliser GNU Débogueur (GDB), qui est portable et fonctionne sur des systèmes de type Unix tels que Linux, ainsi que sur Windows . GDB fonctionne pour de nombreux langages de programmation, notamment Ada, Java, C, C++ et Objective-C. Grâce à GDB, vous pouvez déboguer vos applications localement ou à distance.

Accéder

Dans la fenêtre Analyseur d'Exécution , soit :

- Localisez et double-cliquez sur le script requis et sélectionnez la page ' Déboguer > Plateforme' ou
- Cliquez sur  dans la barre d'outils de la fenêtre, sélectionnez le Paquetage dans lequel créer un nouveau script, et sélectionnez la page ' Déboguer > Plateforme'

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 le Débogueur GNU

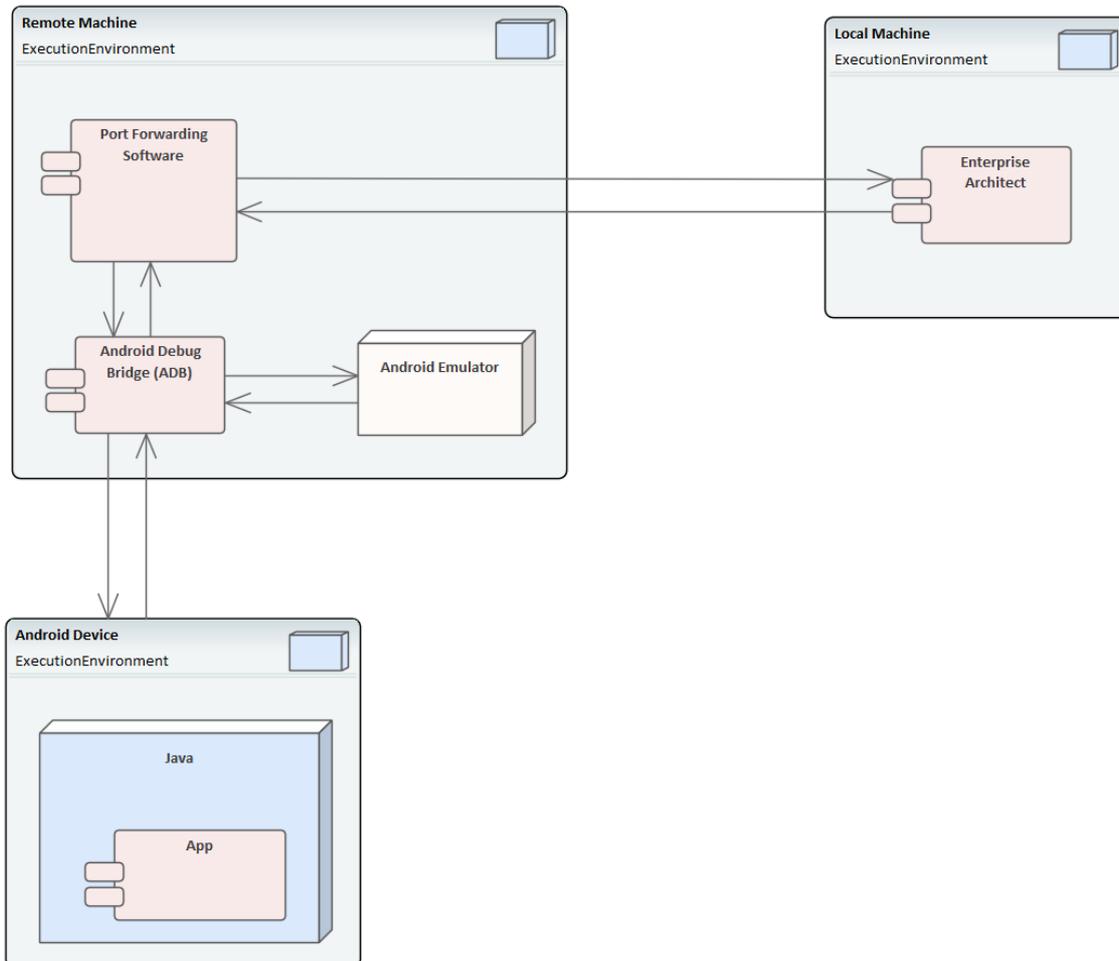
Tâche	Détails
Configurer le script	Un script d'analyse est une exigence de base pour le débogage dans Enterprise Architect ; vous créez un script à l'aide de la barre d'outils Analyseur d'Exécution . Sur la page 'Plateforme' de l'Execution Éditeur de Script Analyseur , dans le champ ' Déboguer ' cliquez sur la flèche déroulante et sélectionnez 'GDB'.
Définir les paramètres de connexion	Le panneau de propriétés affiche un certain nombre de paramètres de connexion pour lesquels vous fournissez des valeurs. <ul style="list-style-type: none"> • path - <path> - Le chemin d'accès complet du fichier exécutable GDB ; vous ne le spécifiez que si GDB ne peut pas être trouvé dans le chemin système • source - <path>, <path> - Le chemin dans lequel le débogueur recherchera les fichiers sources, s'ils ne résident pas dans le répertoire exécutable • remote - F - Définir pour le débogage à distance ; sinon, laisser vide • port - <nnnn> - Le port auquel se connecter sur le serveur distant • hôte - localhost - Le nom de l'hôte auquel se connecter • fetch - T - Définir pour récupérer le binaire à partir du système distant • dumpgdb - <path> - Le nom de fichier dans lequel écrire la sortie GDB • initpath - <path> - Le chemin d'accès complet au fichier gbinit

Notes

- Une exigence de GDB est que le chemin d'accès au fichier de votre code source ne contienne pas d'espaces ; le débogueur ne exécute pas correctement avec des espaces dans le chemin d'accès au fichier

Le Débogueur Android

Si vous développez des applications Java exécutées sur des appareils ou des émulateurs Android, vous pouvez également les déboguer. Les machines locales et distantes peuvent être sur une plate-forme 32 bits ou 64 bits.



Exigences du système

Sur la machine distante, ce logiciel est requis :

- SDK Android, qui comprend le pont de débogage Android, ADB (vous devez être familier avec le SDK et ses outils)
- Java JDK (support 32 et 64 bits)
- Logiciel de redirection de port (tiers)

Sur la machine locale, ce logiciel est requis :

- Enterprise Architect version 10 ou supérieure

Paramètres du script d'analyse

Champ/Bouton	Action

Débogueur	Cliquez sur la flèche déroulante et sélectionnez Java (JDWP).
Exécuter	Cliquez sur ce bouton radio.
Répertoire par défaut	Sans objet – laisser vide.
Parcours d'application	Sans objet – laisser vide.
Arguments de la ligne de commande	Sans objet – laisser vide.
Construire d'abord	Sans objet – laisser vide.
Afficher la console	Sans objet – laisser vide.
Afficher les messages de diagnostic	Sans objet – laisser vide.
Connexion	Sans objet – laisser vide.
Port	Il s'agit du port d'application, attribué via adb ou d'autres moyens, via lequel Enterprise Architect et la Virtual Machine Android (VM) peuvent communiquer.
Hôte	Ordinateur hôte (par défaut, localhost) Si Android s'exécute sur un émulateur sur un appareil connecté à un ordinateur en réseau, saisissez le nom du réseau ici. Par défaut, le débogage tentera de se connecter au port que vous spécifiez sur la machine locale.
Source	Il s'agit de l'équivalent source du paramètre classpath en Java. La racine de chaque arbre source doit être répertoriée. Si plusieurs racines sont spécifiées, elles doivent être séparées par un point-virgule, c'est-à-dire : c:\myapp\src;c:\myserver\src Vous devez spécifier au moins un chemin source racine. Lorsqu'un point d'arrêt se produit, le débogueur recherche la source Java dans chacune des arborescences sources répertoriées ici.
Enregistrement	Permet d'enregistrer des informations supplémentaires à partir du débogueur valeurs possibles : vrai, faux, 1, 0, oui, non
Sortir	Spécifie le nom complet du fichier log local à écrire. Le dossier doit exister sinon aucun log ne sera créé. Le fichier log contient généralement un vidage des octets envoyés entre le débogueur et la machine virtuelle.
Plate-forme	Si vous déboguez Java exécuté dans n'importe quel scénario Android, sélectionnez Android. Pour tous les autres scénarios, sélectionnez Java.

Configurer les ports pour le débogage - Redirection de port (local)

Le débogueur ne peut déboguer qu'une seule machine virtuelle à la fois ; il utilise un seul port pour communiquer avec la machine virtuelle. Le port de l'application à déboguer peut être attribué à l'aide d'ADB, fourni avec le SDK Android.

Avant de procéder au débogage, démarrez l'application une fois sur l'appareil. Lorsque l'application démarre, découvrez son identifiant de processus (pid) :

```
adb jdwp
```

Le dernier numéro répertorié est le PID de la dernière application lancée ; note le PID et utilisez-le pour permettre au débogueur de se connecter à la VM :

- adb transférer tcp:port jdwp:pid
 - port = Numéro de port répertorié dans le script de l'analyseur
 - pid = identifiant de processus de l'application sur l'appareil

Configurer les ports pour le débogage - Redirection de port (à distance)

Pour déboguer à distance, la même procédure doit être suivie que pour la machine locale, mais la communication nécessite une transmission supplémentaire car le socket créé à l'aide de la commande adb forward n'écouterait que sur l'adaptateur local. Le socket est lié à l'hôte local et les tentatives de connexion à ce port se verront confrontées à des messages « connexion refusée ».

Afin de réaliser un débogage à distance, il est nécessaire d'avoir un proxy exécuté sur la machine distante qui écoute toutes les connexions entrantes et transmet tout le trafic au port adb ; de nombreux logiciels sont disponibles pour ce faire.

Le débogage à distance avec Enterprise Architect ne fonctionnera pas si vous n'avez pas configuré un redirecteur de port proxy.

Débogueur Java JDWP

Java propose deux technologies de débogage principales : un système basé sur un agent en cours de processus appelé Java Virtual Machine Tools Interface (JVMTI) et un paradigme basé sur un socket appelé Java Débogueur Wire Protocol (JDWP). Une Virtual Machine Java peut nommer l'un ou l'autre de ces éléments, mais pas les deux, et la fonctionnalité doit être configurée au démarrage de la JVM.

Exigences du système

1. Le débogueur JDWP Enterprise Architect ne pourra communiquer qu'avec une JVM démarrée avec l'option « JDWP ». Voici un exemple d'option de ligne de commande :

```
java -agentlib:jdwp=transport=dt_socket,address=localhost:9000,server=y,suspend=n -cp "c:\java\myapp;%classpath%" demo.myApp "param1" "param2"
```
2. La Virtual Machine ne doit pas être actuellement attachée à un débogueur.
3. Il n'est pas possible qu'une machine virtuelle soit déboguée par Enterprise Architect et Eclipse en même temps.

Paramètres du script d'analyse

Champ/Bouton	Action
Débogueur	Cliquez sur la flèche déroulante et sélectionnez Java (JDWP).
Exécuter	Cliquez sur ce bouton radio pour exécuter le débogueur lorsque le script est exécuté.
Répertoire par défaut	Sans objet – laisser vide.
Parcours d'application	Sans objet – laisser vide.
Arguments de la ligne de commande	Sans objet – laisser vide.
Construire d'abord	Sans objet – laisser vide.
Afficher la console	Sans objet – laisser vide.
Afficher les messages de diagnostic	Sans objet – laisser vide.
Connexion	Sans objet – laisser vide.
Port	Définissez le port d'application attribué au processus VM lors du démarrage, dans les options de ligne de commande Java.
Hôte	Définir l'ordinateur hôte (par défaut, localhost) Si la machine virtuelle s'exécute sur un ordinateur en réseau, entrez le nom du réseau ou l'URL ici. Par défaut, le débogage tentera de se connecter au port que vous spécifiez sur la

	machine locale.
Source	<p>Il s'agit de l'équivalent source du paramètre <i>classpath</i> en Java.</p> <p>Répertoriez la racine de chaque arborescence source ; spécifiez au moins un chemin d'accès à la racine source. Si vous en spécifiez plusieurs, séparez-les par un point-virgule ; par exemple :</p> <p>c:\monapplication\src ; c:\monserveur\src</p> <p>Lorsqu'un point d'arrêt se produit, le débogueur recherche la source Java dans chacune des arborescences sources répertoriées ici.</p>
Enregistrement	<p>Activer ou désactiver la journalisation d'informations supplémentaires du débogueur.</p> <p>Les valeurs possibles incluent :</p> <ul style="list-style-type: none"> • vrai • FAUX • 1 • 0 • Oui • Non
Sortir	<p>Spécifiez le nom complet du fichier log local à écrire. Si le dossier n'existe pas déjà, aucun log ne sera créé.</p> <p>Le fichier log contient généralement un vidage des octets envoyés entre le débogueur et la machine virtuelle.</p>
Plate-forme	Sélectionnez Java.

Configurer les ports pour le débogage

Le débogueur ne peut déboguer qu'une seule machine virtuelle à la fois ; il utilise un seul port pour communiquer avec la machine virtuelle. Le port de l'application à déboguer est attribué lors de la création de la machine virtuelle.

Débogage local

Lorsque Enterprise Architect et la machine virtuelle Java s'exécutent sur la même machine, vous pouvez effectuer un débogage local. Il est nécessaire de lancer la machine virtuelle avec le transport JDWP activé. Consultez la documentation sur *Java Platform Débogueur Architecture (JPDA)* chez Oracle pour connaître les spécifications des options de ligne de commande. Par exemple :

```
java -agentlib:jdwp=transport=dt_socket,address=localhost:9000,server=y,suspend=n -cp
"c:\samples\java\myapp;%classpath%" exemples.MyApp "param1" "param2"
```

Dans cet exemple, les valeurs du script Analyzer seraient « hôte : localhost » et « port : 9000 ».

Débogage à distance

Lorsque Enterprise Architect s'exécute sur la machine locale et que la machine virtuelle Java s'exécute sur une machine

distante, vous pouvez effectuer un débogage à distance. Il est nécessaire de lancer la machine virtuelle avec le transport JDWP activé. Consultez la documentation sur JPDA chez Oracle pour connaître les spécifications des options de ligne de commande. Voici un exemple, où l'ordinateur distant porte le nom de réseau testmachine1 :

```
java -agentlib:jdwp=transport=dt_socket,address=9000,server=y,suspend=n -cp "c:\samples\java\myapp;%classpath%"  
exemples.MyApp "param1" "param2"
```

Note l'absence de nom d'hôte dans l'adresse. Cela signifie que la machine virtuelle écoutera une connexion provenant de n'importe quelle machine. Dans cet exemple, les valeurs du script Analyzer seraient « host: testmachine1 » et « port: 9000 ».

Sortie Point de Trace

La page Tracepoints du script Analyzer vous permet de diriger l'emplacement de la sortie de toutes les instructions Trace au cours d'une session de débogage.

Accéder

Dans la fenêtre Analyseur d'Exécution , soit :

- Localisez et double-cliquez sur le script requis et sélectionnez la page ' Déboguer > Tracepoints ' ou
- Cliquez sur  dans la barre d'outils de la fenêtre, sélectionnez le Paquetage dans lequel créer un nouveau script, et sélectionnez la page ' Déboguer > Tracepoints '.

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

Propriétés Point de Trace

Champ	Détail
Sortir	Vous pouvez choisir entre deux options : <ul style="list-style-type: none"> • 'Ecran' (par défaut) - La sortie est dirigée vers la fenêtre Déboguer • « Fichier » – La sortie est dirigée vers le fichier
Dossier	Entrez le dossier à utiliser pour les fichiers log des instructions Trace.
Nom de fichier	Saisissez le nom à utiliser pour les fichiers log des instructions Trace.
Écraser	Si cette option est sélectionnée, le fichier spécifié est écrasé à chaque démarrage d'une session de débogage.
Numéro automatique	Si cette option est sélectionnée, le fichier log de trace est composé du nom de fichier que vous spécifiez et d'un numéro. Chaque fois que vous démarrez une session de débogage, le numéro est incrémenté.
Sortie de trace de préfixe avec fonction	Si cette option est sélectionnée, toutes les instructions Trace exécutées pendant l'exécution de la session de débogage sont préfixées par l'appel de fonction en cours.

Configuration Établi

Cette rubrique décrit les exigences pour la configuration de l' Object Établi sur Java et Microsoft .NET .

Accéder

Dans la fenêtre Analyseur d'Exécution , soit :

- Localisez et double-cliquez sur le script souhaité et sélectionnez la page ' Déboguer > Établi ' ou
- Cliquez sur  dans la barre d'outils de la fenêtre, sélectionnez le Paquetage dans lequel créer un nouveau script, et sélectionnez la page ' Déboguer > Établi '

Ruban	Develop > Source Code > Analyseur d'Exécution > Edit Scripts d'Analyseur Exécuter > Outils > Analyseur
Raccourcis Clavier	Maj+F12

Plateformes

Plate-forme	Détail
Plateformes prises en charge	L' Établi supporte ces plateformes : <ul style="list-style-type: none"> • Microsoft .NET (version 2.0 ou ultérieure) • Java (JDK 1.4 ou version ultérieure)
Microsoft .NET Établi	L'atelier .NET nécessite un assemblage, qui est utilisé pour créer les éléments de l'atelier. Vous spécifiez le chemin d'accès à l'assembly sur la page ' Établi ' du script Analyzer. Il existe deux contraintes dans l'utilisation de .NET Workbench : <ul style="list-style-type: none"> • Les membres définis comme structure dans le code managé ne sont pas pris en charge • Les classes définies comme internes ne sont pas prises en charge
Java Établi	L'atelier Java utilise les paramètres Virtual Machine configurés dans la page ' Déboguer ' du script Analyzer pour créer la JVM.

Microsoft C++ et natif (C, VB)

Vous ne pouvez déboguer du code natif que s'il existe un fichier PDB correspondant à l'exécutable. Un fichier PDB est créé à la suite de la création de l'application.

La build doit inclure des informations de débogage complètes et aucune optimisation ne doit être définie.

Le script doit spécifier deux éléments pour support le débogage :

- Le chemin vers l'exécutable
- Microsoft Native comme plate-forme de débogage

Configuration générale

Il s'agit de la configuration générale pour le débogage des applications natives Microsoft (C++, C, Visual Basic). Vous avez deux options lors du débogage :

- Déboguer une application
- Se connecter à une application en cours d'exécution

Option 1 – Déboguer une application

Champ	Action
Débogueur	Sélectionnez Microsoft Native comme plate-forme de débogage.
x64	Cochez cette case si vous déboguez une application 64 bits. Décochez la case si vous déboguez une application 32 bits.
Mode	Sélectionnez le bouton radio Exécuter .
Répertoire par défaut	Ceci est défini comme répertoire par défaut pour le processus en cours de débogage.
Chemin d'application	Sélectionnez et entrez le chemin complet ou relatif vers l'exécutable de l'application. <ul style="list-style-type: none"> • Si le chemin contient des espaces, spécifiez le chemin complet ; n'utilisez pas de chemin relatif • Si le chemin contient des espaces, le chemin doit être entouré de guillemets
Arguments de la ligne de commande	Paramètres à passer à l'application au démarrage.
Afficher la console	Créer une fenêtre de console pour le débogueur ; non applicable pour la connexion à un processus.
Chemins de recherche de symboles	Spécifiez tous les chemins supplémentaires pour localiser les symboles de débogage pour le débogueur ; séparez les chemins par un point-virgule.

Option 2 – Se connecter à une application en cours d'exécution

Champ	Action
Débogueur	Sélectionnez Microsoft Native comme plate-forme de débogage.
x64	Cochez cette case si vous déboguez une application 64 bits. Décochez la case si vous déboguez une application 32 bits.

Mode	Sélectionnez le bouton radio Attacher au processus.
Chemins de recherche de symboles	<p>Spécifiez tous les chemins supplémentaires pour localiser les symboles de débogage pour le débogueur.</p> <p>Vous pouvez spécifier un serveur de symboles ici si vous préférez ; séparez les chemins par un point-virgule ou une virgule.</p>

Symboles Déboguer

Pour les applications créées à l'aide de Microsoft Platform SDK, les symboles Déboguer sont écrits dans un fichier PDB d'application lors de la création de l'application.

Les outils de débogage pour Windows, une API utilisée par Visual Execution Débogueur, utilisent ces symboles pour présenter des informations significatives aux contrôles Analyseur d'Exécution.

Ces symboles peuvent facilement devenir obsolètes et provoquer un comportement anormal : le débogueur peut mettre en évidence la mauvaise ligne de code dans l'éditeur alors qu'il se trouve à un point d'arrêt ; il est donc préférable de s'assurer que l'application est créée avant toute session de débogage ou d'enregistrement.

Le débogueur doit informer l'API de la manière de réconcilier les adresses dans l'image en cours de débogage ; il le fait en spécifiant un certain nombre de chemins vers l'API qui lui indiquent où rechercher les fichiers PDB.

Pour les DLL système (kernel32, mfc90ud) pour lesquelles aucun symbole de débogage n'est trouvé, la Pile d'Appel affiche quelques trames avec uniquement les noms et adresses des modules.

Vous pouvez compléter les symboles traduits en passant des chemins supplémentaires à l'API ; vous passez des chemins de symboles supplémentaires dans une liste séparée par des points-virgules dans l'onglet « Déboguer ».

Script de fusion

Une commande de fusion dans un script Analyzer fournit aux utilisateurs une commande supplémentaire pour effectuer une action. L'action de fusion dépend de vos besoins.

Accéder

Dans la fenêtre Analyseur d'Exécution , soit :

- Recherchez et double-cliquez sur le script requis et sélectionnez la page « Fusionner » ou
- Cliquez sur  dans la barre d'outils de la fenêtre, sélectionnez le Paquetage dans lequel créer un nouveau script et sélectionnez la page « Fusionner »

Ruban	Develop > Source Code > Analyseur d'Exécution > Edit Scripts d'Analyseur Exécuter > Source > Fusionner
Raccourcis Clavier	Ctrl+Alt+M

Script Code Miner

Le système Code Miner utilise un ensemble de bases de données pour fournir un accès rapide et complet aux informations issues du code source existant. Les fonctionnalités Intelli-sense des éditeurs de code d' Enterprise Architect et ses outils de recherche peuvent exploiter les informations extraites de ces bases de données.

Grâce aux pages de script Code Miner , vous pouvez spécifier les bases de données Code Miner à utiliser avec un projet particulier et vous pouvez créer, mettre à jour et ajouter de nouvelles bases de données à la bibliothèque Code Miner . La page « Services » vous permet de spécifier une bibliothèque Code Miner locale ou d'indiquer que vous souhaitez accéder à la bibliothèque disponible via le service Intel Sparx.

Différents détails Code Miner peuvent être spécifiés pour chaque script d'analyse, de sorte que les bibliothèques Code Miner utilisées sont déterminées par le script d'analyse *actif* .

Accéder

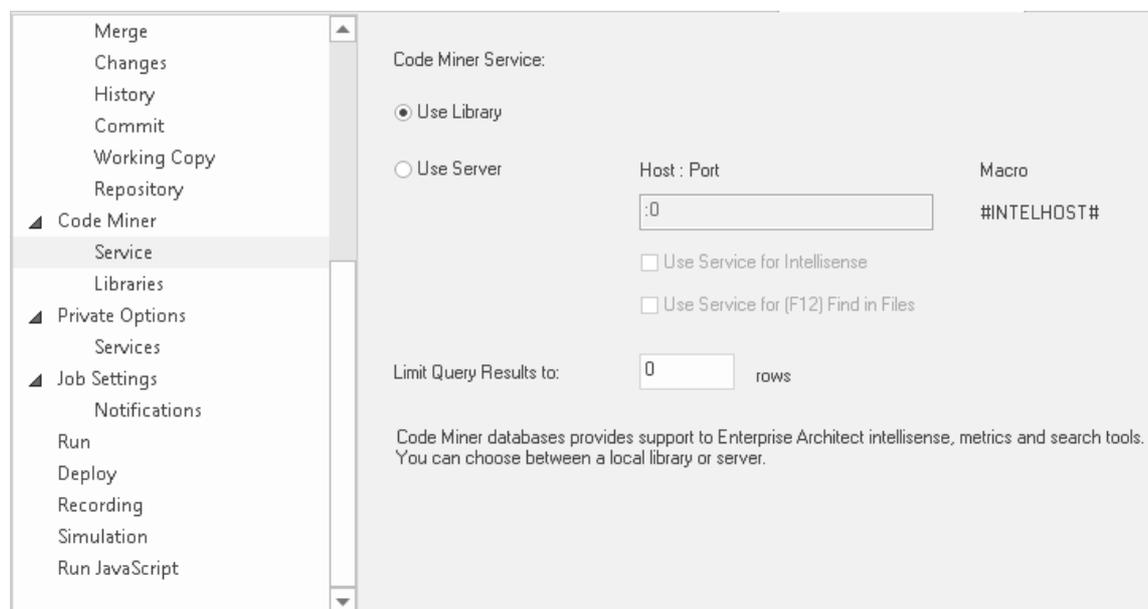
Dans la fenêtre Analyseur d'Exécution , soit :

- Localisez et double-cliquez sur le script requis et sélectionnez la page « Code Miner » ou
- Cliquez sur  dans la barre d'outils de la fenêtre, sélectionnez le Paquetage dans lequel créer un nouveau script et sélectionnez la page « Code Miner »

Ruban	Exécuter > Outils > Analyseur > Vue Scripts d'Analyseur > Double-cliquez sur le nom du script > Code Miner > Service Développer > Source Code > Analyseur d'Exécution > Edit Scripts d'Analyseur > Double-cliquez sur Nom du script > Code Miner > Service
Raccourcis Clavier	Maj+F12

Service Intel Sparx

Une Bibliothèque Code Miner peut être utilisée localement ou déployée sur un serveur où elle peut servir plusieurs clients. Vous sélectionnez le scénario à utiliser sur la page « Service Code Miner » du script Analyzer.

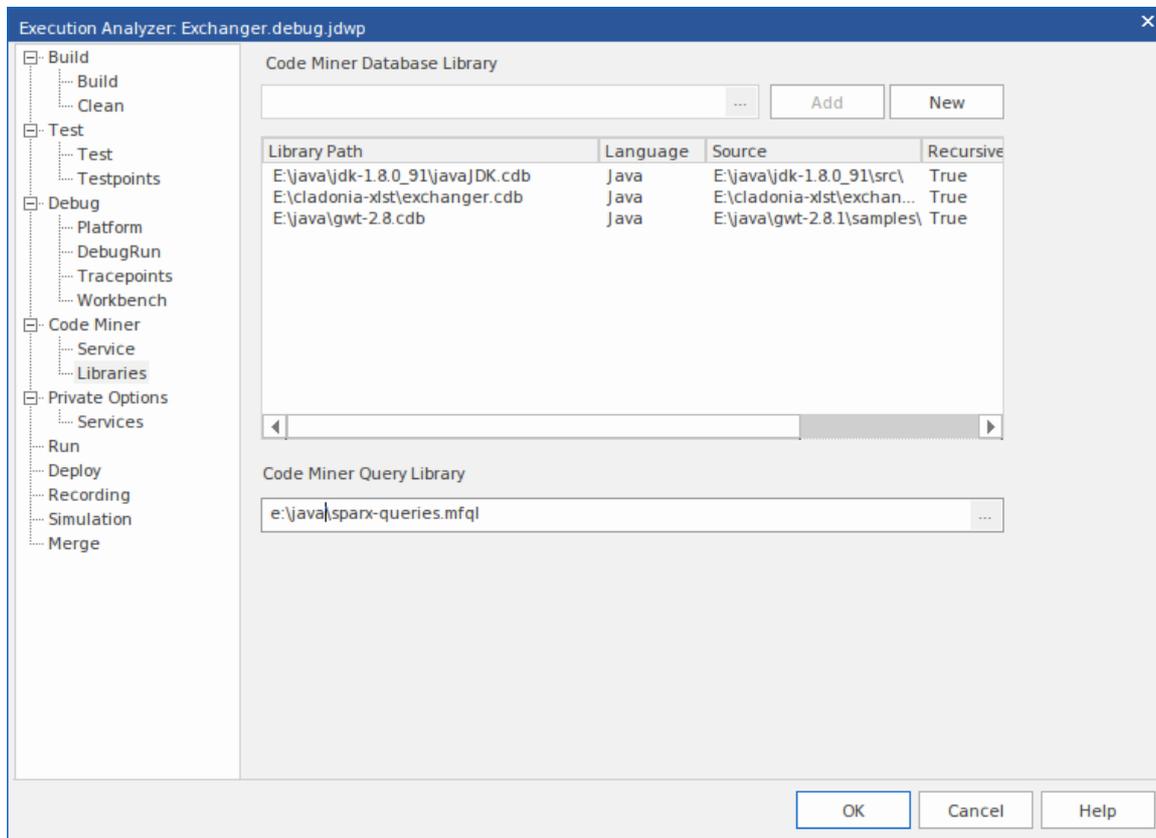


Bibliothèques Code Miner

Les bibliothèques de bases de données Code Miner sont un ensemble de bases de données contenant des informations dérivées du code source. Ensemble, ces bases de données forment la Bibliothèque Code Miner utilisée par les fonctionnalités Intelli-sense d' Enterprise Architect . En général, une bibliothèque est créée pour chaque framework ou projet. La page Bibliothèques Code Miner permet de créer de nouvelles bases de données et d'ajouter, de mettre à jour ou de supprimer des bases de données existantes d'une bibliothèque.

Les bibliothèques Query Code Miner sont une collection de fonctions, écrites dans le langage mFQL de Code Miner , regroupées dans un seul fichier source.

La Bibliothèque de base de données Code Miner et Bibliothèque Query pour un script d'analyse donné sont spécifiées sur la page ' Code Miner | Bibliothèques ' de l' Éditeur de Script .



Script de services

La page 'Services' d'un script Analyzer décrit les ports par défaut utilisés lorsque des scripts sont créés par diverses fonctions Analyseur d'Exécution Visuelle (Importer un projet, Générer Statemachine Exécutable). Vous pouvez mettre à jour n'importe quelle spécification de port sur cette page.

Accéder

Dans la fenêtre Analyseur d'Exécution , soit :

- Recherchez et double-cliquez sur le script requis et sélectionnez la page « Options privées | Services » ou
- Cliquez sur  dans la barre d'outils de la fenêtre, sélectionnez le Paquetage dans lequel créer un nouveau script et sélectionnez la page « Options privées | Services »

Ruban	Develop > Source Code > Analyseur d'Exécution > Edit Scripts d'Analyseur Exécuter > Outils > Analyseur
Raccourcis Clavier	Maj+F12

Exécuter le script

Cette section décrit comment créer une commande pour exécuter votre code exécutable.

Accéder

Dans la fenêtre Analyseur d'Exécution , soit :

- Localisez et double-cliquez sur le script souhaité et sélectionnez la page « Exécuter » ou
- Cliquez sur  dans la barre d'outils de la fenêtre, sélectionnez le Paquetage dans lequel créer un nouveau script, et sélectionnez la page ' Exécuter ' .

Ruban	Develop > Source Code > Analyseur d'Exécution > Edit Scripts d'Analyseur Exécuter > Outils > Analyseur
Raccourcis Clavier	Ctrl+Alt+N

Éléments de script

Élément	Description
Commande	Il s'agit de la commande qui est exécutée lorsque vous sélectionnez l'option du ruban 'Exécuter > Exécuter > Démarrer ' ; dans sa forme la plus simple, le script contiendrait l'emplacement et le nom du fichier à exécuter .
Exemples	Ces deux exemples montrent des scripts configurés pour exécuter une application .Net et une application Java dans Enterprise Architect . .File: C:\benchmark\cpp\exemple_net_1\version\exemple.exe Java: client La commande répertoriée dans ce champ est exécutée comme si elle provenait de l'prompt de commande ; par conséquent, si le chemin de l'exécutable ou des arguments contiennent des espaces, ils doivent être placés entre guillemets.

Notes

- Enterprise Architect offre la possibilité de démarrer votre application normalement OR avec un débogage à partir du même script ; le menu « Analyseur » propose des options distinctes pour démarrer un exécuter normal et un exécuter de débogage

Script de déploiement

Ces sections expliquent comment créer un script de commande pour déployer le Paquetage actuel. Le script peut être exécuté en sélectionnant l'option de ruban « Exécuter > Source > Créer > Déployer » ou en appuyant sur Ctrl+Maj+Alt+F12.

Accéder

Dans la fenêtre Analyseur d'Exécution , soit :

- Recherchez et double-cliquez sur le script requis et sélectionnez la page « Déployer » ou
- Cliquez sur  dans la barre d'outils de la fenêtre, sélectionnez le Paquetage dans lequel créer un nouveau script et sélectionnez la page « Déployer »

Ruban	Develop > Source Code > Analyseur d'Exécution > Edit Scripts d'Analyseur Exécuter > Outils > Analyseur
Raccourcis Clavier	Ctrl+Maj+Alt+F12

Actes

Action	Détail
Exécutez la commande en tant que :	<p>Process</p> <p>If the deployment is handled externally, enter the path to the program or batch file to run, followed by any parameters; the program is launched in a separate process.</p> <p>Example:</p> <pre>C:\apache-ant-1.7.1\bin\ant.cmd myproject deploy</pre> <p>Batch File</p> <p>When using this option, you can enter multiple commands that are then executed as a single script in a command console; you have access to any environment variables available in a standard command console.</p> <p>Example:</p> <pre>@echo on IF NOT EXIST "%1%" GOTO DEPLOY_NOWAR IF "%APACHE_HOME%" == "" GOTO DEPLOY_NOAPACHE xcopy /L "%1%" "%APACHE_HOME%\webapps" GOTO DEPLOY_END rem rem NO WAR FILE rem :DEPLOY_NOWAR echo "%1% WAR file not found"</pre>

	<pre>GOTO DEPLOY_END rem rem NO APACHE ENVIRONMENT VARIABLE rem :DEPLOY_NOAPACHE echo "APACHE_HOME environment variable not found" :DEPLOY_END pause</pre>
Analyser la sortie	<p>La sélection d'un Parser dans la liste entraîne la capture de la sortie du script de déploiement ; la sortie est analysée selon la syntaxe sélectionnée dans la liste.</p> <p>Pour afficher la fenêtre Sortie système, sélectionnez l'option de ruban « Démarrer > Toutes Windows > Conception > Explorer > Système ».</p>

Scripts Enregistrement

L'intérêt de l'enregistrement n'est pas tant de nous permettre d'avoir une vue d'ensemble, mais plutôt de pouvoir voir une image plus petite qui a une part de vérité à dire. Nous avons tous vu diagrammes Séquence qui ne sont pas très utiles. (*Le même message apparaissant 100 fois de suite sur un diagramme nous dit quelque chose, mais pas grand-chose.*) Heureusement, Enterprise Architect prend en charge ce premier point grâce à l'utilisation de fragments. Les comportements répétitifs sont identifiés comme Motifs et représentés une fois sous forme de fragment sur le diagramme Séquence. Le fragment est étiqueté en fonction du nombre d'itérations. L'historique d'enregistrement, bien sûr, montre toujours l'historique complet. Nous avons également besoin d'outils pour nous aider à concentrer l'enregistrement sur des domaines d'intérêt particuliers et à réduire le bruit des autres. Nous pouvons utiliser des filtres pour ce faire. Avec les filtres, vous pouvez exclure toutes les classes, fonctions ou même modules de tout enregistrement. Vous pouvez créer plusieurs ensembles de filtres et les utiliser avec des ensembles de marqueurs pour cibler différents cas d'utilisation.

Accéder

Dans la fenêtre Analyseur d'Exécution, soit :

- Localisez et double-cliquez sur le script requis et sélectionnez la page « Enregistrement » ou
- Cliquez sur  dans la barre d'outils de la fenêtre, sélectionnez le Paquetage dans lequel créer un nouveau script, et sélectionnez la page ' Enregistrement '

Ruban	Develop > Source Code > Analyseur d'Exécution > Edit Scripts d'Analyseur Exécuter > Outils > Analyseur
Raccourcis Clavier	Maj+F12

Chaînes de filtrage

Élément	Discussion
Filtration	<p>Si la case à cocher « Activer le filtre » est sélectionnée sur la page « Enregistrement » de l'Execution Éditeur de Script Analyseur, le débogueur exclut les appels aux méthodes correspondantes de l'enregistrement. La comparaison est sensible à la casse.</p> <p>Pour ajouter une valeur, cliquez sur l'icône « Nouveau » (« Insérer ») dans le coin droit de la case « Filtres d'exclusion » et saisissez la string de comparaison ; chaque string de filtre prend la forme :</p> <pre>nom_de_classe_jeton::nom_de_méthode_jeton</pre> <p>Le class_name_token exclut les appels à toutes les méthodes d'une ou plusieurs classes dont le nom correspond au jeton ; la string peut contenir le caractère générique * (astérisque).</p> <p>La méthode method_name_token exclut les appels aux méthodes dont le nom correspond au jeton ; encore une fois, la string peut contenir le caractère générique *.</p> <p>Les deux jetons sont facultatifs ; si aucun jeton de classe n'est présent, le filtre est appliqué uniquement aux fonctions globales ou publiques (c'est-à-dire aux méthodes n'appartenant à aucune classe).</p>

Exemple	<p>Dans cet exemple Java, le débogueur exclurait :</p> <ul style="list-style-type: none"> • Appels à la méthode OnDraw pour la classe Example.common.draw.DrawPane • Appels à n'importe quelle méthode de n'importe quelle classe dont le nom commence par Example.source.Collection • Appels à n'importe quel constructeur pour n'importe quelle classe (comme <clint> et <init>) <div data-bbox="520 421 970 573" style="border: 1px solid gray; padding: 5px;"> <p>Filters</p> <p>Example.common.draw.DrawPane::OnDraw</p> <p>Example.source.Collection*</p> <p>*:init*</p> </div> <p>Dans cet exemple de code natif, le débogueur exclurait :</p> <ul style="list-style-type: none"> • Appels effectués vers l'espace de noms Standard Gabarit Bibliothèque • Appels à n'importe quelle classe commençant par TOB • Appels à n'importe quelle méthode de la classe CLock • Appels à la méthode GetLocation pour la classe CTrain • Appels à toute fonction globale ou publique dont le nom commence par Get <div data-bbox="520 875 970 1099" style="border: 1px solid gray; padding: 5px;"> <p>Filters</p> <p>std*</p> <p>TOB*</p> <p>CLock</p> <p>CTrain::GetLocation</p> <p>::Get*</p> </div>
---------	--

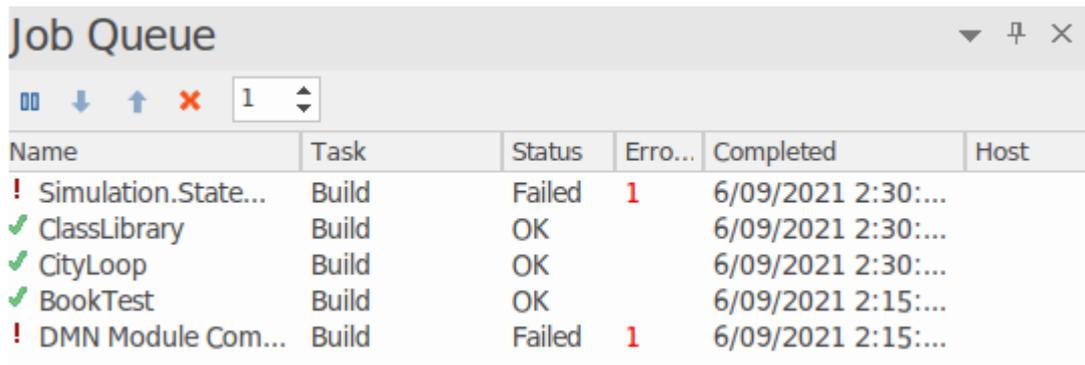
Filtres

Utiliser l'entrée de filtre	Pour filtrer
::Get*	Toutes les fonctions publiques dont le nom commence par « Get » proviennent de la session d'enregistrement (par exemple, GetClientRect dans l'API Windows).
::Get	Toutes les méthodes commençant par « Get » dans n'importe quelle classe.
CClass::Get*	Toutes les méthodes commençant par Get pour la classe CClass.
Classe C::*	Toutes les méthodes pour la classe CClass.
ATL* mst*	Toutes les méthodes pour les classes appartenant aux bibliothèques Gabarit standard et Actif Gabarit .
CClass::GetName	La ou les méthodes spécifiques GetName pour la classe CClass.

La fenêtre de la file d'attente des tâches

La fenêtre File d'attente des tâches simplifie le processus de travail avec Scripts d'Analyseur, qui étaient à l'origine traités individuellement et si aucun autre script n'était exécuté. Dans les versions d' Enterprise Architect à partir de la version 16.0, lorsqu'une option de menu contextuel de script d'analyse est exécutée (par exemple, « Build »), elle est placée dans une file d'attente de tâches ; plusieurs tâches peuvent être mises en file d'attente et d'autres tâches peuvent être effectuées pendant que les tâches sont traitées.

Le nom du script Analyzer est utilisé comme nom du travail. Un script Analyzer peut avoir plusieurs sections, telles que Build, Test, Exécuter et Deploy, et chaque section est affectée en tant que tâche du travail.



Name	Task	Status	Erro...	Completed	Host
! Simulation.State...	Build	Failed	1	6/09/2021 2:30:...	
✓ ClassLibrary	Build	OK		6/09/2021 2:30:...	
✓ CityLoop	Build	OK		6/09/2021 2:30:...	
✓ BookTest	Build	OK		6/09/2021 2:15:...	
! DMN Module Com...	Build	Failed	1	6/09/2021 2:15:...	

La sortie de chaque tâche exécutée à partir de la fenêtre File d'attente des tâches est capturée dans l'onglet « Historique des tâches » de la fenêtre Sortie système.

Accéder

Ruban	Exécuter > Outils > Analyseur > File d'attente des tâches Vue Démarrer > Toutes Windows > Conception > Explorer > Système > Historique des tâches
-------	--

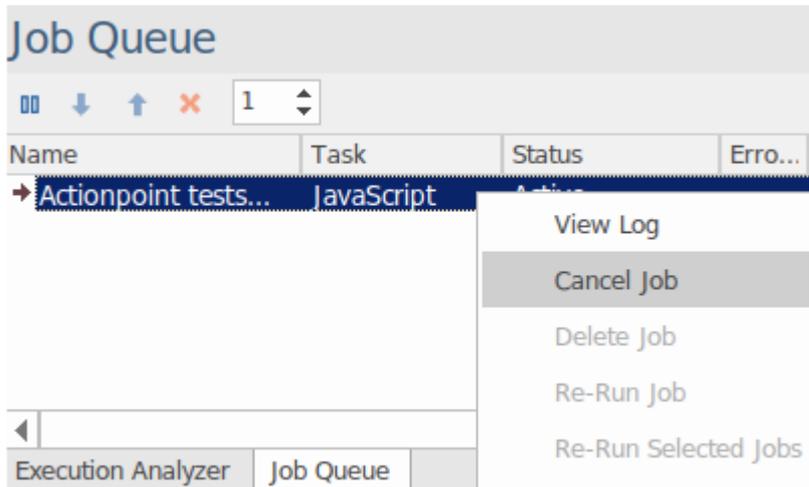
Colonnes de la file d'attente des tâches

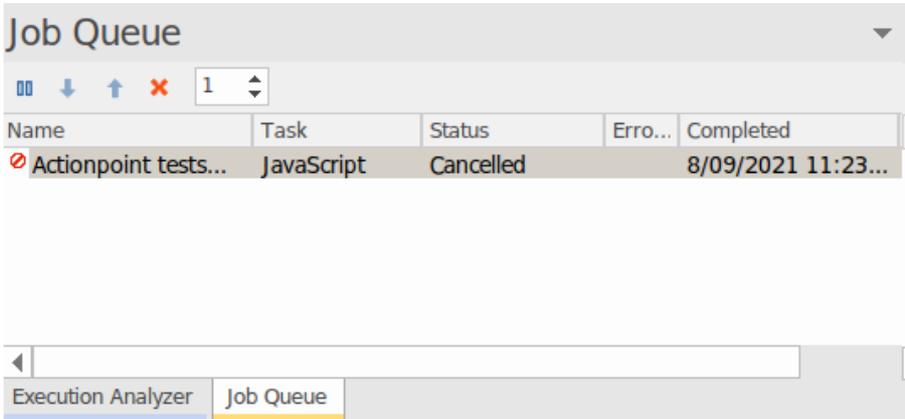
Colonne	Description
Nom	Nom du script d'analyse pour lequel le travail a été ajouté à la file d'attente des travaux. Si le travail a été exécuté, le nom sera précédé d'une coche (pour une exécution réussie) ou d'un point d'exclamation (pour un échec).
Tâche	La section du script d'analyse que le travail exécute, par exemple, Build ou Deploy.
Statut	L'état d'achèvement du travail : si le travail s'est terminé avec succès (« OK ») ou a échoué.
Erreurs	Si le travail a été exécuté et a échoué, le nombre d'erreurs survenues.
Complété	La date et l'heure à laquelle le travail a été terminé.

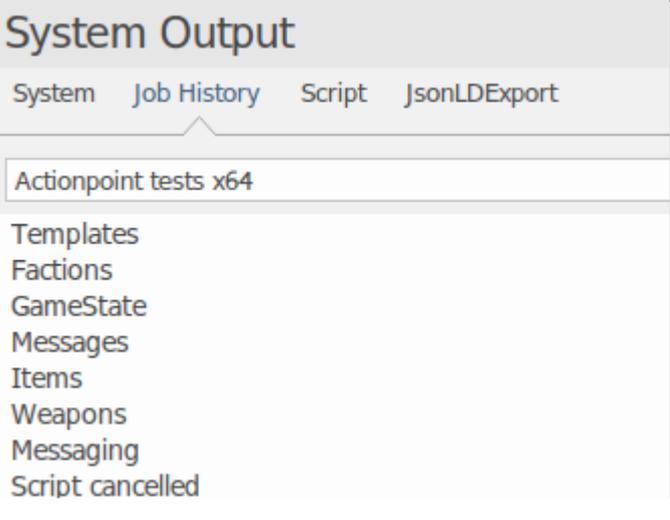
Hôte	Si le travail est exécuter à distance, l'adresse IP ou le nom d'hôte de la machine distante.
Reçu	Si le travail est exécuter à distance, tout message de retour de la machine hôte.

Options Menu Contexte

Cliquez-droit sur un nom de job, ou sur l'arrière-plan de la fenêtre, pour afficher le menu contextuel 'Job Queue'.



Option	Description
Journal Vue	Sélectionnez cette option pour afficher l'onglet « Historique des tâches » de la fenêtre Sortie système, pour une tâche qui a été exécutée.
Annuler le travail	<p>Cliquez-droit sur un nom de tâche et cliquez sur cette option, si nécessaire et à tout moment, pour annuler la tâche JavaScript sélectionnée.</p> <p>L'annulation est indiquée dans la fenêtre File d'attente des tâches par une icône « aucune entrée » à côté du nom de la tâche.</p>  <p>The screenshot shows the 'Job Queue' window with a table containing job information. The job 'Actionpoint tests... JavaScript' is now in the 'Cancelled' status. The 'Completed' column shows the date and time '8/09/2021 11:23...'. The window title is 'Job Queue' and it has several control icons at the top.</p> <p>Dans l'onglet « Historique des tâches » de la fenêtre Sortie système, la sortie est terminée avec un message « Script annulé ».</p>

	
Supprimer le travail	Cliquez sur une tâche qui n'a pas encore démarré et sélectionnez cette option pour la supprimer de la file d'attente des tâches.
Réexécuter le travail	Sélectionnez cette option pour exécuter à nouveau la tâche terminée sélectionnée.
Réexécuter les tâches sélectionnées	(Ctrl+clic sur un certain nombre de tâches requises.) Sélectionnez cette option pour exécuter à nouveau toutes les tâches terminées sélectionnées.
Réexécuter les tâches terminées	Sélectionnez cette option pour exécuter à nouveau toutes les tâches terminées dans la liste actuelle.

L'onglet Historique des tâches

La sortie de chaque tâche exécutée à partir de la fenêtre File d'attente des tâches est capturée dans l'onglet « Historique des tâches » de la fenêtre Sortie système. À partir de là, vous pouvez visualiser n'importe quel log de tâches à votre guise, en le sélectionnant dans la liste déroulante de la barre d'outils. Si la tâche a échoué et qu'il y a des messages d'erreur, vous pouvez passer d'un message à l'autre à l'aide des icônes de flèche rouge. Ces icônes sont désactivées s'il n'y a aucun message d'erreur actif.

System Output

System Job History Script JsonLDExport

Actionpoint tests x64

```

1>----- Build started: Project: ActionPoints, Configuration: Debug x64 -----
1>stdafx.cpp
1>ActionPoints.cpp
1>ActionPoints.vcxproj -> C:\EA\Stephen\ActionPoints\x64\Debug\ActionPoints.exe
===== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped =====
process ended, exitcode 0
Completed Build section
Build succeeded
Started Deploy section
Executing script on remote host 192.168.20.9:9000
Connecting to 192.168.20.9:9000...
Connected.

```

Options de la barre d'outils de la file d'attente des tâches

Option	Description
	Cliquez sur un nom de travail et cliquez sur cette icône pour mettre en pause ou reprendre le travail sélectionné.
	Cliquez sur un nom de tâche et sur l'une de ces deux flèches pour déplacer la tâche vers le haut ou vers le bas dans la file d'attente des tâches, la plaçant plus tôt ou plus tard dans l'ordre de traitement.
	Cliquez sur le nom d'une tâche pour une tâche qui n'a pas encore démarré, et sur cette icône pour supprimer cette tâche de la fenêtre File d'attente des tâches.
	Cliquez sur la flèche vers le haut ou vers le bas pour définir le nombre de tâches pouvant exécuter simultanément, jusqu'à un maximum de 8. Le nombre par défaut est de 1 afin que la file d'attente des travaux traite les travaux un par un, premier entré, premier sorti.

Créer une application

Cette rubrique explique comment exécuter un script Build sur votre application, dans Enterprise Architect .

Accéder

Ruban	Exécuter > Source > Construire > Construire
Raccourcis Clavier	Ctrl+Maj+F12
Autre	Barre d'outils « Construire » >  Fenêtre Analyseur d'Exécution 

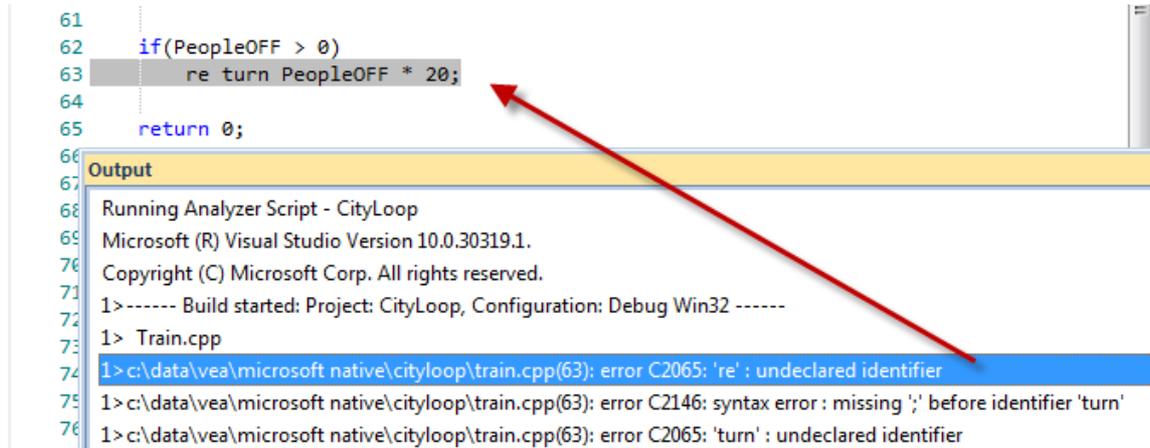
Action

Lorsque vous sélectionnez l'option « Build », la commande « Build » du script sélectionné dans la fenêtre Analyseur d'Exécution est exécutée. La progression et le résultat de l'opération de build sont affichés dans l'onglet « Build » de la fenêtre Sortie système.

Vous pouvez rapidement visiter la ligne de code pour toute erreur de compilation apparaissant en double-cliquant sur l'erreur.

Localiser les erreurs Compilateur dans le code

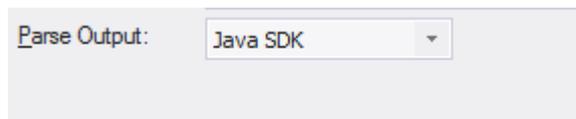
Lorsque vous créez une application à l'aide d'un script Analyzer, la sortie du compilateur est enregistrée dans la fenêtre Sortie système. Vous pouvez double-cliquer sur n'importe quel message d'erreur qui apparaît ici et accéder au code source. Lorsque vous faites cela, le curseur est positionné sur la ligne contenant l'erreur.



```
61
62     if(PeopleOFF > 0)
63         re turn PeopleOFF * 20;
64
65     return 0;
66
67 Output
68 Running Analyzer Script - CityLoop
69 Microsoft (R) Visual Studio Version 10.0.30319.1.
70 Copyright (C) Microsoft Corp. All rights reserved.
71 1>----- Build started: Project: CityLoop, Configuration: Debug Win32 -----
72 1> Train.cpp
73 1> c:\data\vea\microsoft native\cityloop\train.cpp(63): error C2065: 're' : undeclared identifier
74 1> c:\data\vea\microsoft native\cityloop\train.cpp(63): error C2146: syntax error : missing ';' before identifier 'turn'
75 1> c:\data\vea\microsoft native\cityloop\train.cpp(63): error C2065: 'turn' : undeclared identifier
76
```

Conseil

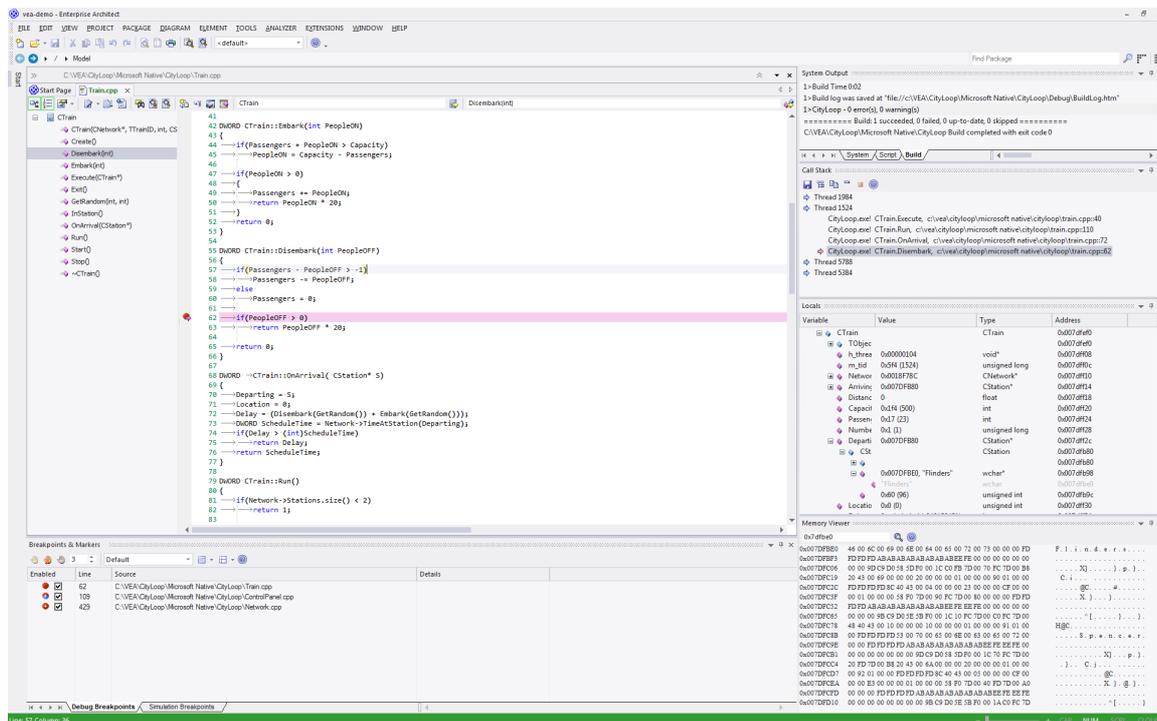
Si la sortie est manquante, vérifiez qu'un analyseur de langue est mentionné dans le script de l'analyseur (Maj+F12).



Accéder

Ruban	Démarrer > Toutes Windows > Conception > Explorer > Système
Raccourcis Clavier	Ctrl+Maj+8

Débogage



Enterprise Architect est bien plus qu'un simple outil de dessin : il offre également toutes fonctionnalités que vous pouvez attendre d'un IDE. Des environnements et des outils de débogage complets pour de nombreuses plateformes majeures sont mis à disposition. L'intégration de la capacité de débogage dans l'outil de modélisation permet au code d'être développé, construit et géré par ses auteurs. Travailler et collaborer dans un modèle intégré a permis de rendre les actions importantes et chaque action responsable d'une manière qui n'est tout simplement pas possible avec d'autres chaînes d'outils.

Fonctionnalités

Vitesse

Les débogueurs d'Enterprise Architect sont rapides ! L'exécution des programmes ne vous prendra pas toute la journée. Le programme d'enregistrement peut être exécuté sans intervention manuelle.

Support

- C++, C et Visual Basic
- Microsoft .NET, ASP.NET, WCF
- Java, utilisant le transport de socket (JDWP) ou le modèle en mémoire (JVMT)
- Android sur un émulateur ou un appareil
- JavaScript, VBScript et JScript
- Scripts PHP sur les serveurs Web Apache
- Processus GDB Linux distants à l'aide Enterprise Architect sous Windows
- Simulation - débogage des simulations en UML et BPMN
- Statemachines Exécutables - déboguer une Statemachine en cours d'exécution

Isolement

Les débogueurs fonctionnent hors processus d'Enterprise Architect, isolant ainsi des effets secondaires.

Efficacité

Le démarrage et l'arrêt du débogueur sont rapides et simples. Cela ne vous retient pas. Conçu pour être une UI réactive, le thread principal UI est isolé des tâches qui ne relèvent pas de sa responsabilité.

Productivité

Passez de modélisation aux exigences, de la création d'une demande de modification au suivi des modifications de code dans un modèle partagé au sein d'une organisation, en passant par le profilage des modifications de code récentes. Le tout dans un seul outil.

Notes

- Les fonctionnalités de débogage et d'enregistrement de l'Analyseur d'Exécution Visuelle ne sont pas disponibles pour la plateforme serveur Java 'Weblogic' d'Oracle.

Exécuter le Débogueur

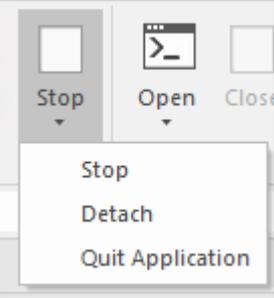
Enterprise Architect propose plusieurs méthodes pour démarrer et contrôler une session de débogage. Il existe la fenêtre principale Débogueur, ainsi qu'une barre d'outils Débogueur et le panneau « Exécuter » dans le ruban « Exécuter ». Il est toujours préférable d'afficher la fenêtre Débogueur lorsque vous exécutez une session de débogage, car c'est là que toutes les sorties de débogage sont capturées.

Accéder

Ruban	Exécuter > Exécuter > Démarrer Exécuter > Outils > Débogueur > Démarrer le débogage
Raccourcis Clavier	Alt+8 (affiche la fenêtre Débogueur) F6 (commence l'exécution de l'application en cours de débogage)
Barre d'outils	Explorer > Portails > Afficher la barre d'outils > Débogueur

Utilisation de la fenêtre Débogueur

Action	Détail
Démarrer le Débogueur	<p>Lorsqu'un script Analyzer a été configuré pour support le débogage, vous pouvez démarrer le débogueur de ces manières :</p> <ul style="list-style-type: none"> • Dans le ruban, sélectionnez 'Exécuter > Exécuter > Démarrer > Exécuter '. • Depuis le ruban, sélectionnez « Exécuter > Outils > Débogueur > Démarrer Debugging » • Dans la barre d'outils ' Débogueur ', cliquez sur le bouton , ou • Appuyez sur F6 <p>Vous pouvez également lancer le débogueur pour n'importe quel script via son menu contextuel dans la « Fenêtre de script de l'analyseur » ou appuyer sur Maj+F12</p> <p>Si vous n'avez pas de script Analyzer, il est toujours possible de déboguer une application en cours d'exécution en vous connectant directement à ce processus :</p> <ul style="list-style-type: none"> • Depuis le ruban, sélectionnez « Exécuter > Outils > Débogueur > Attacher au processus », ou • Dans la barre d'outils ' Débogueur ', cliquez sur le bouton  (Attacher) et choisissez manuellement la plateforme de débogage
Pause/Reprise du débogage	<p>Vous pouvez suspendre une session de débogage ou reprendre la session après une pause de ces manières :</p> <ul style="list-style-type: none"> • Depuis le ruban, sélectionnez « Exécuter > Exécuter > Pause » • Dans la barre d'outils ' Débogueur ', cliquez sur le bouton 

Arrêtez le Débogueur	<p>Le débogueur se termine normalement lorsque le processus de débogage en cours se termine. Cependant, certaines applications et certains services (tels que Java Virtual Machine) peuvent nécessiter l'arrêt manuel du débogueur. Pour arrêter le débogage, procédez comme suit :</p> <ul style="list-style-type: none"> • Dans la barre d'outils ' Débogueur ', cliquez sur le bouton  (Stop) • Appuyez sur Ctrl+Alt+F6 • Sélectionnez la flèche déroulante sur l'option du ruban « Exécuter > Exécuter > Arrêter » <p>L'option ruban affiche un menu court proposant trois manières de terminer le débogage de l'application.</p>  <ul style="list-style-type: none"> • Arrêter - arrête le Débogueur et arrête le processus en cours de débogage (valeur par défaut lorsque vous cliquez simplement sur l'icône du ruban) • Détacher - arrête le Débogueur mais laisse le processus en cours d'exécution • Quitter l'application - arrête le Débogueur et affiche un message WM_QUIT dans la fenêtre principale du processus, s'il en a une
Parcourir les lignes de code	<p>Pour passer à la ligne de code suivante :</p> <ul style="list-style-type: none"> • Depuis le ruban, sélectionnez « Exécuter > Exécuter > Pas à pas », ou • Dans la barre d'outils ' Débogueur ', cliquez sur le bouton  (Pas à pas), ou • Appuyez sur Alt+F6
Entrez dans Appels de Fonction	<p>Pour accéder à un appel de fonction :</p> <ul style="list-style-type: none"> • Dans le ruban, sélectionnez 'Execute > Exécuter > Entrer ', ou • Dans la barre d'outils ' Débogueur ', cliquez sur le bouton  (Entrer), ou • Appuyez sur Maj+F6 <p>Si aucune source n'est disponible pour la fonction cible alors le Débogueur revient immédiatement à l'appelant.</p>
Sortir les fonctions	<p>Pour quitter une fonction :</p> <ul style="list-style-type: none"> • Dans le ruban, sélectionnez 'Execute > Exécuter > Sortir ' • Dans la barre d'outils ' Débogueur ', cliquez sur le bouton  (Sortir), ou • Appuyez sur Ctrl+F6 <p>Si le débogueur entre dans une fonction sans code source, il continuera à sortir jusqu'à ce qu'un point contenant du code source soit trouvé.</p>
Afficher le point d'exécution	<p>Pendant que le Débogueur est en pause, pour revenir au fichier source et à la ligne de code que le Débogueur est sur le point d'exécuter :</p> <ul style="list-style-type: none"> • Dans le ruban, sélectionnez 'Exécuter > Exécuter > Démarrer > Afficher le point d'exécution'.

	<ul style="list-style-type: none"> • Dans la barre d'outils ' Déboguer ', cliquez sur le bouton  (Afficher le point d'exécution). <p>La ligne appropriée est mise en évidence, avec une flèche rose dans la marge gauche de l'écran.</p>
Sortir	<p>Lors d'une session de débogage, des messages s'affichent dans la fenêtre Déboguer détaillant :</p> <ul style="list-style-type: none"> • Démarrage de la session • Fin de session • Exceptions • Erreurs • Messages Déboguer trace, tels que ceux générés à l'aide de Java System.out ou .NET System.Diagnostics. <p>Si vous double-cliquez sur un message de débogage, soit :</p> <ul style="list-style-type: none"> • Une fenêtre contextuelle s'affiche avec un texte de message plus complet, ou • S'il y a eu une fuite de mémoire, le fichier est affiché à l'endroit où l'erreur s'est produite
Enregistrer la sortie (et Effacer la sortie)	<p>Vous pouvez enregistrer l'intégralité du contenu de la sortie Déboguer dans un fichier .txt externe ou enregistrer les lignes sélectionnées de la sortie dans le presse-papiers Enterprise Architect .</p> <p>Pour enregistrer toutes les sorties dans un fichier, cliquez sur le bouton  (Enregistrer la sortie dans un fichier).</p> <p>Pour enregistrer les lignes sélectionnées dans le presse-papiers, cliquez-droit sur la sélection et sélectionnez l'option « Copier la sélection dans le presse-papiers ».</p> <p>Lorsque vous avez enregistré la sortie ou que vous ne souhaitez plus l'afficher, cliquez-droit sur la sortie actuelle et sélectionnez l'option ' Effacer les résultats'.</p>
Passer au profileur	<p>Si vous exécutez une session de débogage sur du code, vous pouvez arrêter la session de débogage et passer immédiatement à une session de profilage.</p> <p>Pour passer du Déboguer au Profiler :</p> <ul style="list-style-type: none"> • Depuis le ruban, sélectionnez « Exécuter > Outils > Déboguer > Basculer vers le profileur » • Dans la fenêtre Déboguer , cliquez sur l'option '  Switch to Profiler', ou • Dans la barre d'outils Déboguer , cliquez sur l'option '  Switch to Profiler' <p>Le profileur s'attache au processus en cours d'exécution.</p> <p>Cette facilité n'est pas disponible pour les débogueurs Java.</p>

Gestion des Point d'Arrêt et Marqueurs

Points d'Arrêt fonctionnent dans Enterprise Architect de la même manière que dans n'importe quel autre Débogueur . Marqueurs sont similaires aux points d'arrêt, mais dans Enterprise Architect ils ont des pouvoirs spéciaux. En termes simples, les marqueurs effectuent des actions - telles que l'enregistrement de l'exécution et l'analyse - que les points d'arrêt ne font pas. L'action d'un point d'arrêt consiste toujours à arrêter le programme.

Vous définissez n'importe quel marqueur ou point d'arrêt dans l'éditeur de code source, où ils sont visibles dans la marge de gauche. Cliquer dans cette marge ajoutera un point d'arrêt sur cette ligne. Points d'Arrêt et les marqueurs sont interchangeables - vous pouvez changer un point d'arrêt en marqueur et vice versa, en utilisant sa dialogue « Propriétés ». Vous pouvez rapidement afficher et modifier les propriétés d'un point d'arrêt ou d'un marqueur en utilisant Ctrl+clic soit sur son icône dans la marge de l'éditeur, soit dans la fenêtre Points d'Arrêt et Marqueurs .

Points d'Arrêt sont conservés dans des ensembles. Il existe un ensemble par défaut pour chaque modèle et chaque point d'arrêt y réside généralement, mais vous pouvez enregistrer la configuration actuelle des points d'arrêt sous forme d'ensemble nommé, créer un nouvel ensemble et basculer entre eux. Les ensembles Point d'Arrêt sont partagés, c'est-à-dire qu'ils sont disponibles pour la communauté des modèles. L'exception est l'ensemble par défaut qui est un ensemble privé et personnel alloué à chaque utilisateur de n'importe quel modèle.

Accéder

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

Options Point d'Arrêt et Marqueur

Option	Détail
Supprimer un point d'arrêt ou un marqueur	<p>Pour supprimer un point d'arrêt spécifique :</p> <ul style="list-style-type: none"> • Si le point d'arrêt est activé, cliquez sur le cercle rouge du point d'arrêt dans la marge gauche de la Source Éditeur de Code , ou • Cliquez-droit sur le point d'arrêt ou le marqueur dans la Source Éditeur de Code , le dossier <i>Points d'Arrêt</i> ou la fenêtre Points d'Arrêt & Marqueurs et sélectionnez l'option 'Supprimer', ou • Sélectionnez le point d'arrêt dans l'onglet ' Débogueur Points d'Arrêt ' et appuyez sur la touche Suppr
Supprimer tous les points d'arrêt	Cliquez sur le bouton Supprimer tous les points d'arrêt ().
Propriétés Point d'Arrêt	Dans la fenêtre Points d'Arrêt ou dans l'éditeur de code, utilisez le menu contextuel du marqueur pour afficher les propriétés. Vous pouvez ici modifier le type de marqueur, ajouter ou modifier des contraintes et saisir des instructions de trace. (Raccourci utile : maintenez la touche Ctrl enfoncée tout en cliquant sur le marqueur pour afficher rapidement ses propriétés.)

Désactiver un point d'arrêt	Décochez la case en regard du point d'arrêt ou du marqueur.
Activer un point d'arrêt ou un marqueur	Cochez la case en regard du point d'arrêt ou du marqueur.
Désactiver tous les points d'arrêt	Cliquez sur le bouton 
Activer tous les points d'arrêt	Cliquez sur le bouton Activer tous les points d'arrêt ().
Interruption lorsque l'adresse mémoire est modifiée	Cliquez sur le bouton Point d'arrêt des données ().
Identifier ou modifier le jeu de marqueurs	Vérifiez le champ <input type="text" value="Default"/> dans la barre d'outils de la fenêtre Points d'Arrêt & Événements . Si nécessaire, cliquez sur la flèche déroulante et sélectionnez un autre ensemble de marqueurs. L'ensemble par défaut est normalement utilisé pour le débogage et est personnel à votre ID utilisateur ; les autres ensembles de marqueurs sont partagés entre tous les utilisateurs du modèle.
Modifier la manière dont les points d'arrêt et les marqueurs sont regroupés dans la fenêtre Points d'Arrêt et Événements	Les points d'arrêt et les marqueurs peuvent être groupés par classe ou par fichier de code. Pour grouper les éléments, cliquez sur la flèche vers le bas de l'icône  dans la barre d'outils, puis cliquez sur l'option appropriée. Si vous ne souhaitez pas grouper les éléments, cliquez sur l'option sélectionnée pour la désélectionner ; les points d'arrêt et les marqueurs sont alors listés par numéro de ligne.

States Point d'Arrêt

State	Remarques
	<i>Déboguer Running</i> : Lié <i>Déboguer non exécuté</i> : activé
	<i>Déboguer Running</i> : Désactivé <i>Déboguer ne fonctionne pas</i> : désactivé
	<i>Déboguer Running</i> : Non lié - cela signifie généralement qu'un module n'a pas encore été chargé. De plus, les dll sont déchargées de temps en temps. <i>Déboguer non exécuté</i> : N/a
	<i>Déboguer Running</i> : Failed - cela signifie que le débogueur n'a pas pu faire correspondre cette ligne de code à une instruction dans l'un des modules chargés. Peut-être que la source provient d'un autre projet ou que la configuration du projet est obsolète. Note que si la date du module est antérieure à la date du code source du point d'arrêt, vous verrez une notification dans la fenêtre du débogueur. Le texte est de couleur rouge pour qu'il ressorte. C'est un signe clair que le projet doit être

	compilé. <i>Déboguer non exécuté : N/a</i>
--	---

Définition Points d'Arrêt du code

Points d'Arrêt normaux sont généralement définis sur une ligne de code source. Lorsque le Débogueur atteint la ligne indiquée pendant l'exécution normale, le Débogueur arrête l'exécution et affiche les variables locales, la pile d'appels, les threads et d'autres informations d'exécution.

Définir un point d'arrêt sur une ligne de code

Étape	Action
1	Ouvrez le code source à déboguer dans l'éditeur de code source intégré.
2	<p>Recherchez la ligne de code appropriée et cliquez dans la colonne de marge de gauche - un cercle rouge uni dans la marge indique qu'un point d'arrêt a été défini à cette position.</p> <pre>12 CTest::CTest(LPCTSTR name, TTestType type) 13 { 14 m_Name = name; 15 m_Type = type; 16 theTest = this; 17 }</pre> <p>Si le code est actuellement arrêté à un point d'arrêt, ce point est indiqué par une flèche bleue à côté du marqueur.</p> <pre>6 int _tmain(int argc, _TCHAR* argv[]) 7 { 8 CTest Test(_T("Model"), CTest::Regression); 9 return Test.Run(); 10 }</pre> <p>Alternativement, vous pouvez définir le marqueur Point d'Arrêt (ou autre marqueur) en cliquant avec le bouton droit sur la marge gauche de la ligne requise, pour afficher le menu contextuel du point d'arrêt/marqueur ; sélectionnez le type de marqueur approprié.</p>

Déclarations de trace

Une instruction de trace est un message qui est généré lors de l'exécution d'une session de débogage. Les instructions de trace peuvent être définies dans Enterprise Architect sans nécessiter de modifications du code source de votre application.

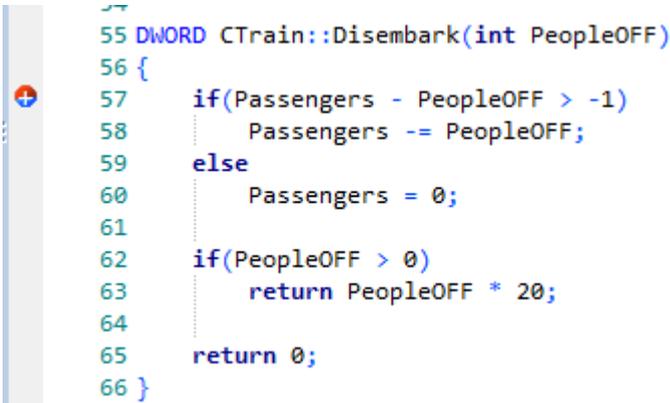
Marqueurs Point de Trace sont définis dans l'éditeur de code. Comme les points d'arrêt, ils sont placés sur une ligne de code. Lorsque cette ligne de code s'exécute, le débogueur évalue l'instruction, dont le résultat est enregistré dans la fenêtre Débogueur (ou dans un fichier s'il est remplacé par le script Analyzer).

Accéder

Toutes les instructions Trace existantes peuvent être visualisées et gérées dans la fenêtre Points d'Arrêt & Marqueurs . La fenêtre Points d'Arrêt & Marqueurs peut être affichée à l'aide de l'une des méthodes décrites ici.

Ruban	Exécuter > Windows > Points d'Arrêt
-------	-------------------------------------

Ajouter un Point de Trace Marqueur

Étape	Action
1	Ouvrez le code source pour déboguer dans l'éditeur de code source.
2	Recherchez la ligne de code appropriée, cliquez-droit dans la marge de gauche et sélectionnez l'option « Ajouter Point de Trace Marqueur ». Si un marqueur est déjà présent, appuyez sur Ctrl+clic pour afficher la fenêtre Propriétés Point d'Arrêt .
3	Assurez-vous que la case à cocher « Trace statement » est sélectionnée.
4	Dans le champ de texte sous la case à cocher « Instruction de trace », saisissez l'instruction de trace requise.
5	<p>Cliquez sur le bouton OK . Un Point de Trace Marqueur s'affiche dans la marge gauche de l'éditeur de code.</p>  <pre> 55 DWORD CTrain::Disembark(int PeopleOFF) 56 { 57 if(Passengers - PeopleOFF > -1) 58 Passengers -= PeopleOFF; 59 else 60 Passengers = 0; 61 62 if(PeopleOFF > 0) 63 return PeopleOFF * 20; 64 65 return 0; 66 } </pre>

Spécification d'une instruction de trace

Une instruction de trace peut être n'importe quel texte libre. La valeur de toutes les variables actuellement dans la portée peut également être incluse dans une instruction de trace en préfixant le nom de la variable avec un jeton spécial.

Les jetons disponibles sont :

- `$` - lorsque la variable doit être interprétée comme une string
- `@` - lorsque la variable est un type primitif (int , double, char)

En utilisant notre exemple dans l'image, nous pourrions afficher le nombre de personnes descendant d'un train en utilisant cette instruction :

Il y avait `@Passengers` avant que `@PeopleOFF` ne descende du train à la station `$Arriving.Name`

En plus de tracer les valeurs des variables à partir de votre code, vous pouvez utiliser les mots-clés `$stack` et `$frame` dans votre instruction Trace pour imprimer la trace de la pile actuelle ; utilisez :

- `$stack` - pour imprimer toutes les images, ou
- `$frame[start](count)` - imprime un nombre spécifique d'images de la pile à partir d'une image donnée ; par exemple, `$frame[0](5)` imprimera l'image actuelle et 4 ancêtres

Notes

- Les instructions de trace peuvent être incluses sur n'importe quel type de point d'arrêt ou de marqueur.

Interrompre lorsqu'une Variable Change de Valeur

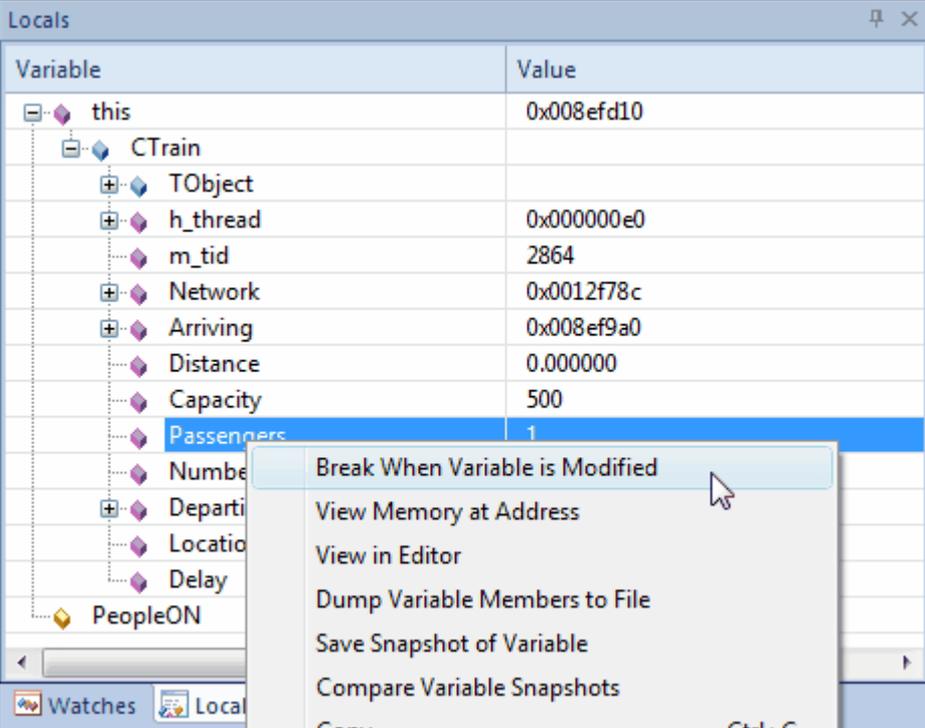
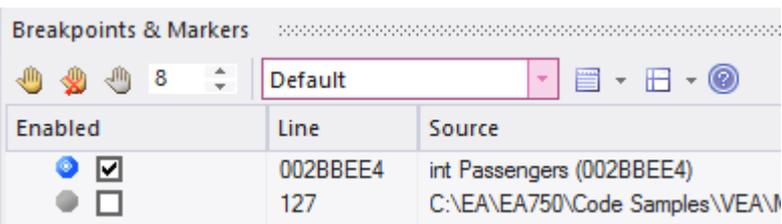
Les points d'arrêt de données peuvent être définis sur une variable mémoire prédéterminée pour forcer le débogueur à arrêter l'exécution à la ligne de code qui vient de provoquer la modification de la valeur de la variable. Cela peut être utile lorsque vous essayez de retrouver le point auquel une variable est modifiée pendant l'exécution du programme, en particulier si l'on ne sait pas clairement comment l'exécution du programme affecte un état object particulier.

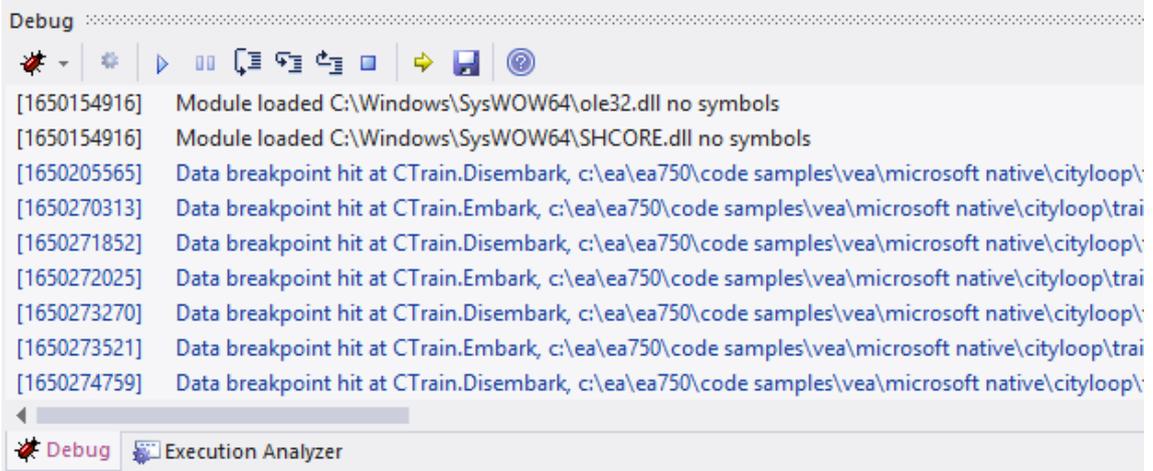
Accéder

Ruban	Exécuter > Windows > Variables Locales : Cliquez-droit sur la variable > Arrêter lorsque la variable est modifiée ou Exécuter > Windows > Observateurs : Cliquez-droit sur la variable > Break When Variable is Modified
Autre	Dans une fenêtre d'éditeur de code : Cliquez-droit sur la variable qui vous intéresse Arrêt lorsque l'élément est modifié

Capturer les modifications apportées à une variable à l'aide de points d'arrêt de données

Mesures	Détail
1	Définissez un point d'arrêt normal dans le code pour pouvoir choisir une variable. Puis exécuter le débogueur (F6).
2	Lorsque le programme a atteint le point d'arrêt, sélectionnez la variable qui vous intéresse et, dans son menu contextuel, sélectionnez l'option « Arrêter lorsque la variable est modifiée ».

	
3	<p>Il n'y a pas d'indicateurs de points d'arrêt dans le code, mais les points d'arrêt de données sont facilement reconnaissables dans la fenêtre Points d'Arrêt et Événements , sous la forme d'une icône bleue avec un losange blanc. Enterprise Architect affiche le nom de la variable et son adresse au lieu d'un numéro de ligne.</p> 
4	<p>Avec le point d'arrêt de données défini, vous pouvez désactiver tous les autres points d'arrêt que vous pourriez avoir. Le programme s'arrêtera à toute ligne de code qui modifie valeur de cette variable. exécuter maintenant votre programme.</p>
5	<p>Lorsque cette variable est modifiée, le débogueur s'arrête et affiche la ligne de code en cours dans l'éditeur. Il ne s'agit pas de la ligne qui a provoqué la coupure, mais de la ligne de code qui suit l'événement. L'événement est enregistré dans la fenêtre Débogueur .</p>



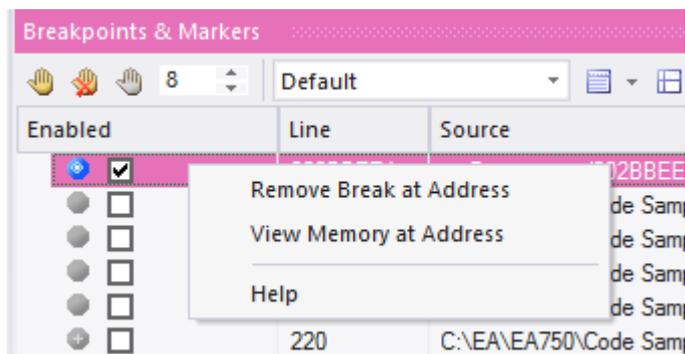
Nous savons maintenant comment et où cette valeur (son State) a changé. Par exemple, l'instruction à la ligne 58 vient de mettre à jour le nombre de passagers.

```

55 DWORD CTrain::Disembark(int PeopleOFF)
56 {
57     if(Passengers - PeopleOFF > -1)
58         Passengers -= PeopleOFF;
59     else
60         Passengers = 0;
61
62     if(PeopleOFF > 0)
63         return PeopleOFF * 20;
64
65     return 0;
66 }

```

- 6 Après avoir découvert cet endroit et d'autres endroits où cette valeur est modifiée, assurez-vous de supprimer la notification avant de continuer. Vous pouvez supprimer rapidement le point d'arrêt des données en le sélectionnant dans la fenêtre Points d'Arrêt et en appuyant sur la touche Supprimer. Vous pouvez également utiliser le menu contextuel cliquer-droit pour ce faire.



Notes

- Cette fonctionnalité n'est actuellement pas prise en charge par la plateforme Microsoft .NET

Trace lorsque Variable Change de Valeur

Lors de l'exécution de votre code, il est possible que la valeur d'une variable soit modifiée. Il est possible de capturer ces modifications et la nouvelle valeur de la variable, dans la fenêtre Déboguer . Vous pouvez ensuite double-cliquer sur l'enregistrement de modification pour afficher la ligne de code qui a provoqué la modification, dans l' Éditeur de Code .

Accéder

Ruban	Exécuter > Windows > Variables Locales : Cliquez-droit sur la variable > Trace lorsque la variable est modifiée ou Exécuter > Windows > Observateurs : Cliquez-droit sur la variable > Trace lorsque la variable est modifiée
Autre	Dans Éditeur de Code Cliquez-droit sur variable Trace lorsque la variable est modifiée

Configurer Trace

La variable que vous tracez doit être dans la portée. Pour l'identifier et la sélectionner, définissez un point d'arrêt normal sur la ligne de code où vous savez que la variable existera. Lorsque le débogueur atteint ce point d'arrêt, localisez la variable et utilisez son menu contextuel pour activer le suivi.

Pour localiser une variable :

- Si vous voyez la variable dans le code source, passez la souris dessus, cliquez-droit et sélectionnez l'option « Afficher la variable » ; Enterprise Architect la localisera
- Si la variable est dans la portée (une variable locale, ou « this » ou un membre de « this »), recherchez-la dans la fenêtre Variables locales ('Exécuter > Windows > Variables locales')
- Si la variable est globale (C, C++), affichez la fenêtre Observateurs ('Exécuter > Windows > Observateurs ') et recherchez-la par son nom
- Si la variable est un membre statique de classe, affichez la fenêtre Observateurs ('Exécuter > Windows > Observateurs ') et saisissez son nom complet

Une fois la trace activée, vous pouvez désactiver tous les autres points d'arrêt et laisser le programme exécuter . Chaque fois que la variable change valeur , elle sera enregistrée dans l'onglet 'Sortie' du débogueur. Vérifiez le changement de valeur et double-cliquez sur la ligne pour afficher le code dans l' Éditeur de Code .

Notes

- Le débogueur ne s'arrête pas lorsque l'événement de modification se produit, il enregistre uniquement la modification
- Ce facilité est disponible sur les plateformes Microsoft Native et Java
- Microsoft .NET ne prend pas support les points d'arrêt sur les valeurs

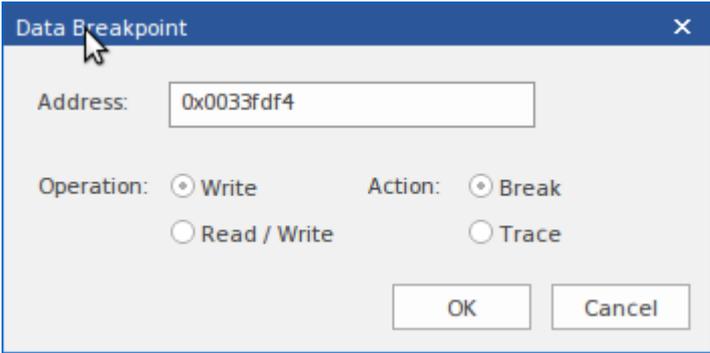
Détection des Opérations d'Adresse Mémoire

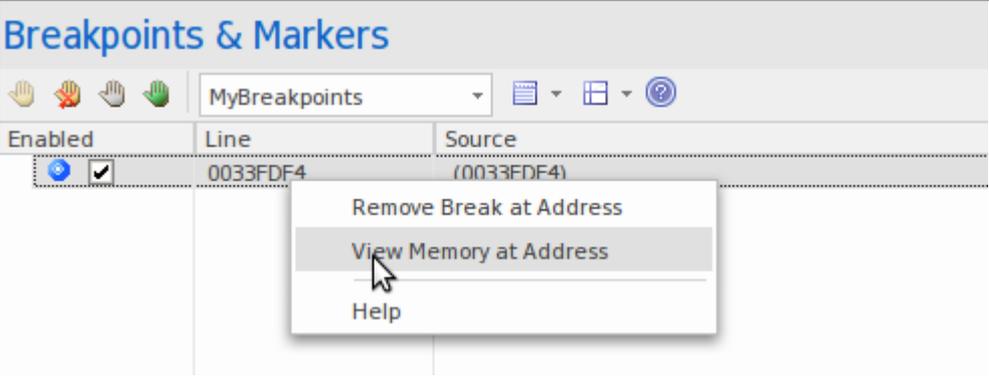
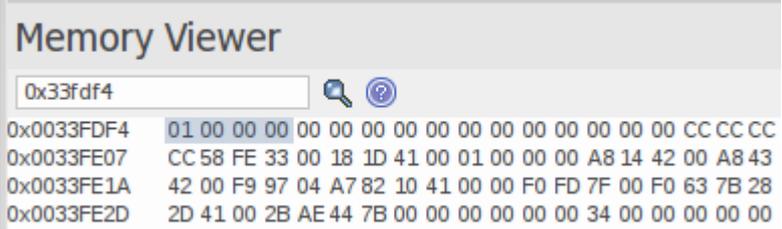
Être capable de détecter où et quand une zone de mémoire est lue ou écrite peut être d'une grande aide pour les enquêteurs, même lorsque la base de code est bien comprise. Sans cet outil, un développeur C++ pourrait avoir la tâche potentiellement ardue de suivre où et quand une variable globale est consultée et de déboguer ces fonctions. Les points d'arrêt de données permettent à un programmeur C++ de suivre quand une variable / un emplacement de mémoire est lu ou quand il est écrit. Lorsque l'opération est détectée, le débogueur arrête l'exécution et la ligne de code suivant l'opération s'affiche dans l'éditeur de code.

Accéder

Ruban	Exécuter > Windows > Points d'Arrêt
-------	-------------------------------------

Détecter l'opération sur l'adresse mémoire

Étape	Action
1	Cliquez sur le bouton  .
2	Entrez l'adresse mémoire à surveiller. Vous pouvez copier une adresse depuis la fenêtre Variables locales. 
3	Sélectionnez l'opération à détecter. Si vous sélectionnez « Écriture », le débogueur s'arrêtera lorsque l'adresse sera écrite. Si vous choisissez « Lecture/Écriture », le débogueur vous avertira lorsque l'adresse sera lue ou lorsqu'elle sera écrite.
4	Sélectionnez l'action à effectuer. Si vous choisissez « Interrompre », le débogueur arrêtera le programme et la ligne de code sera affichée dans l'éditeur. Si vous choisissez « Tracer », le débogueur n'arrêtera pas l'exécution, mais log toute opération sur l'adresse au fur et à mesure qu'elle se produit. Cette sortie est affichée dans la fenêtre Débogueur .
5	Le point d'arrêt des données est ajouté à la fenêtre Points d'Arrêt et Marqueurs .

	
6	<p>Vous pouvez utiliser le menu contextuel sur le point d'arrêt des données pour vérifier la valeur à l'adresse mémoire.</p> 
7	<p>Pour supprimer un point d'arrêt de données, sélectionnez-le dans la fenêtre Points d'Arrêt et Marqueurs et appuyez sur la touche Suppr. Vous pouvez également décocher la case à côté. Les points d'arrêt de données sont supprimés lorsqu'ils sont désactivés ; ils ne persistent pas comme les autres points d'arrêt.</p>

Exigences du système

Les points d'arrêt d'adresse mémoire sont pris en charge dans le débogueur natif C/C++.

Point d'Arrêt Propriétés

Points d'Arrêt possèdent un certain nombre de propriétés supplémentaires qui déterminent ce qui se produit lors de l'exécution de la ligne de code à laquelle le point d'arrêt s'applique.

Ces propriétés définissent :

- L'action à effectuer
- La ligne de code à laquelle le point d'arrêt s'applique
- Contraintes qui déterminent si l'action est exécutée ou non lorsque le point d'arrêt est atteint
- Informations de trace à afficher lorsque le point d'arrêt est atteint

Breakpoint Properties [X]

Action: ● Break Execution Line number:

Constraints:

Trace statement

Accéder

Il existe plusieurs manières d'afficher la dialogue ' Point d'Arrêt Propriétés ' :

Éditeur de Code	<ul style="list-style-type: none"> • Cliquez-droit sur un marqueur de point d'arrêt Propriétés ou • Ctrl+Clic sur le marqueur de point d'arrêt ou • Cliquez-droit sur le code comportant un marqueur de point d'arrêt Point d'Arrêt Propriétés
Vitrine Points d'Arrêt & Marqueurs	<ul style="list-style-type: none"> • Cliquez-droit sur le point d'arrêt Propriétés

Options

Champ	Détails
Action	Le comportement lorsque le point d'arrêt est atteint.

Doubler	La ligne de code source à laquelle ce point d'arrêt s'applique.
Hauteur de la pile	Pour les marqueurs de capture de pile, le nombre d'images d'appelant à enregistrer. Pour enregistrer la pile entière, définissez la valeur sur 0.
Contraintes	Définit la condition dans laquelle l'action du point d'arrêt sera exécutée. Pour les points d'arrêt normaux, il s'agirait de la condition qui interrompt l'exécution. Dans cet exemple, pour un point d'arrêt normal, l'exécution s'arrêterait à cette ligne lorsque la condition est évaluée à True. Les contraintes sont évaluées à chaque fois que la ligne de code est exécutée. <code>(this.m_FirstName="Joe") AND (this.m_LastName="Smith")</code>
Déclaration de trace	Un message s'affiche dans la fenêtre Déboguer lorsque le point d'arrêt est atteint. Les variables actuellement dans la portée peuvent être incluses dans une sortie d'instruction de trace en préfixant le nom de la variable avec un jeton \$ pour les variables string, ou un jeton @ pour les types primitifs tels que int ou long. Par exemple : Le compte \$pAccount->m_sName a un solde de @pAccount->m_fBalance

Défaut de lier Point d'Arrêt

Un échec de point d'arrêt se produit s'il y a un problème de liaison du point d'arrêt. Les échecs Point d'Arrêt sont le plus souvent causés par des fichiers sources modifiés sans que l'application ne soit reconstruite. Points d'Arrêt peuvent parfois être liés à une ligne différente, ce qui entraîne leur déplacement. Si un point d'arrêt ne peut pas être lié au binaire sur cette ligne ou sur les trois lignes qui la suivent, il est affiché avec un point d'interrogation.

Un message d'avertissement s'affiche dans la colonne « Détails » de la fenêtre Points d'Arrêt & Événements , identifiant le type de problème :

- Le fichier source du point d'arrêt ne correspond pas au fichier source utilisé pour créer l'image de l'application
- L'horodatage du fichier est supérieur à celui de l'image

Un message d'avertissement est également affiché dans la fenêtre Déboguer .

Déboguer une Application en Cours

Plutôt que de démarrer un processus explicitement depuis Enterprise Architect, vous souhaitez peut-être déboguer une application (processus) qui est déjà en cours d'exécution sur votre système.

Dans ce cas, vous pouvez utiliser la fonction de débogage pour vous connecter au processus déjà en cours d'exécution. À condition que vous ayez les informations de débogage appropriées écrites dans le processus en cours d'exécution et/ou les fichiers de débogage associés (tels que les fichiers .PDB), le débogueur se connecte à ce processus et lance une session de débogage.

Vous pouvez également vous « détacher » du processus après avoir terminé votre inspection et laisser le processus exécuter normalement.

Accéder

Ruban	Exécuter > Exécuter > Démarrer > Attacher au processus
Fenêtre de Débogueur	La barre d'outils de la fenêtre du débogueur comporte un bouton Attacher

Étapes

Scène	Description
Afficher Processus	Lorsque vous choisissez de déboguer un autre processus, la dialogue « Attacher au processus » s'affiche. Vous pouvez limiter les processus affichés à l'aide des boutons radio en haut de le dialogue ; pour trouver un service tel qu'Apache Tomcat ou ASP.NET, sélectionnez le bouton radio Système.
Sélectionnez Débogueur	Lorsque vous sélectionnez un processus, vous devrez peut-être choisir le débogueur dans la liste déroulante Débogueur ; cependant, si le Paquetage sélectionné a déjà été configuré dans un script d'analyse, le débogueur répertorié dans le script est prédéfini dans le dialogue .
Sélection du processus	Une fois que vous avez double-cliqué sur un processus contenant des informations de débogage et Enterprise Architect est attaché au processus : <ul style="list-style-type: none"> • Tous les points d'arrêt rencontrés sont détectés par le débogueur • Le processus est interrompu lorsqu'un point d'arrêt est rencontré, et • Les informations sont disponibles dans la fenêtre Débogueur
Se détacher du processus	Pour vous détacher d'un processus, cliquez sur le bouton  (Arrêt Débogage).

Voir les Variables Locales

La fenêtre Variables locales affiche les variables du système en cours d'exécution. Que vous enregistriez C# , que vous déboguez du Java, du C++ ou du VBScript, que vous déboguez un Statemachine Exécutable ou que vous exécutiez une simulation, cette fenêtre est l'endroit où se trouvent les variables du système. Les valeurs actuelles ne s'affichent que lorsqu'un programme est arrêté. Cela se produit lorsqu'un point d'arrêt est rencontré pendant le débogage, lorsque vous passez par-dessus une ligne de code ou lorsque vous passez d' States à l'autre dans une simulation.

Accéder

Ruban	Exécuter > Windows > Variables locales Simuler > Simulation Dynamique > Variables locales
Menu Contexte	Dans Éditeur de Code Cliquez-droit sur n'importe quel identifiant de variable > Afficher la variable

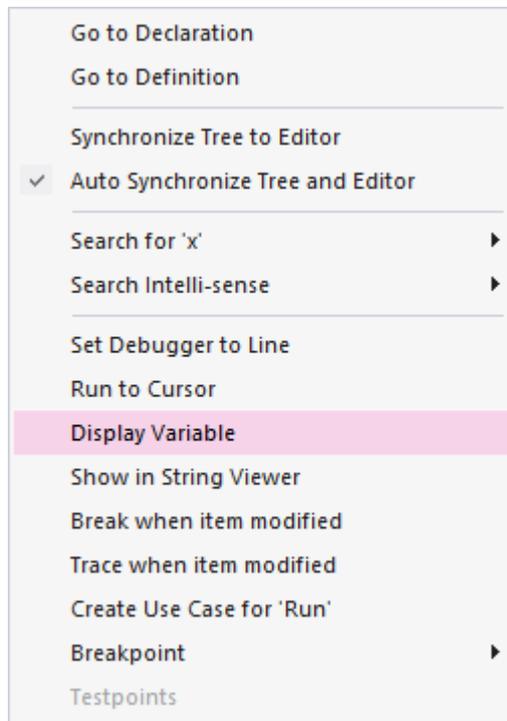
Icônes

La valeur et le type de toute variable concernée sont affichés dans une arborescence ; chaque variable possède une icône de case colorée qui identifie le type de variable :

- Bleu - Object avec membres
- Vert – Tableaux
- Rose - Types élémentaires
- Jaune - Paramètres
- Rouge - Instance Établi

Recherche de variables

La manière la plus simple de trouver une variable est de la localiser d'abord dans l'éditeur de code et d'utiliser le menu contextuel cliquez-droit sur la variable, en sélectionnant « Afficher la variable ». Enterprise Architect recherchera et révélera toute variable dans la portée, y compris les membres profondément imbriqués. Si la variable se trouve dans une portée différente (globale, fichier, module, statique), elle sera affichée dans la fenêtre Observateurs (voir *Voir Variables dans d'Autres Portées*).



Vue persistante

L'examen des variables implique généralement de fouiller dans l'arborescence pour exposer les valeurs intéressantes. Il peut alors être ennuyeux, après avoir traversé cette difficulté, de passer à la ligne de code suivante, seulement pour voir ces variables à nouveau cachées à cause d'un changement de contexte. La fenêtre Variables locales a une vue persistante qui persiste pendant un certain temps après une commande exécuter ou step. Lorsque vous parcourez une fonction dans Enterprise Architect, la structure des variables persiste ligne après ligne. Cela rend le parcours d'une fonction rapide et facile.

Qu'est-ce qui a changé

Dans le cadre de la vue persistante, la fenêtre Variables locales suit les modifications apportées aux valeurs et les met en évidence.

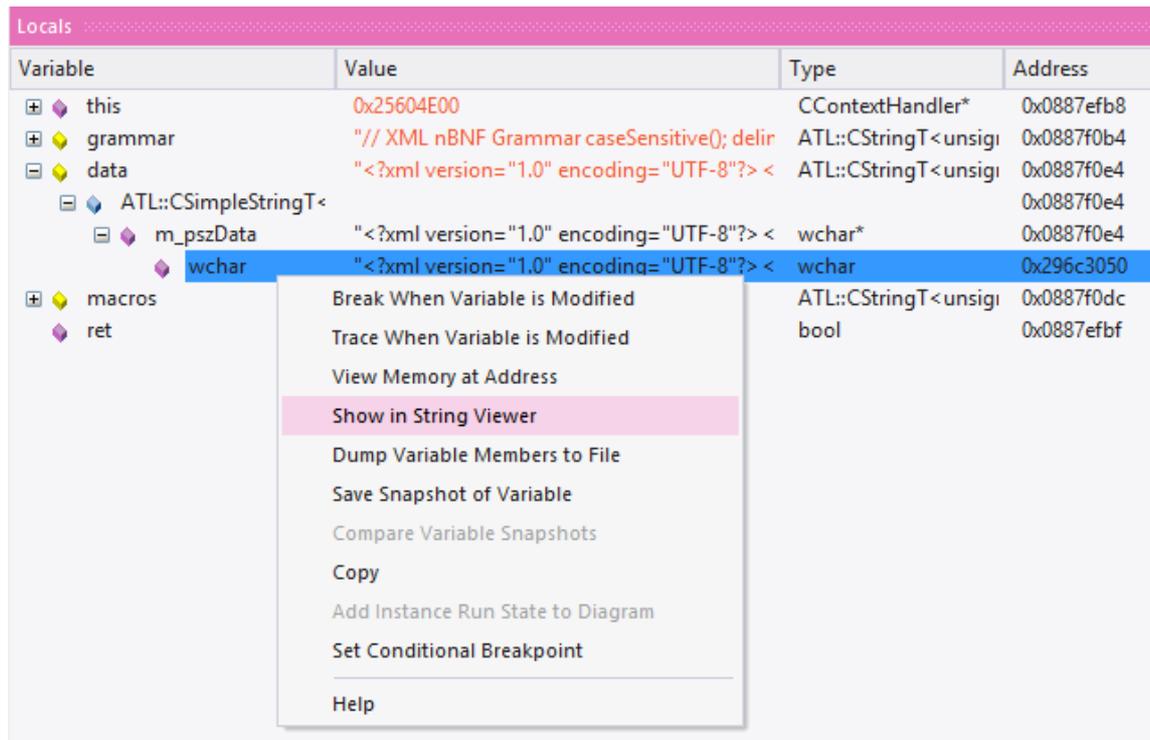
Variable	Value	Type	Address
[-] this	0x02BD0AA0	Exchange::Account*	0x00c8f6d0
[-] Exchange::Account		Exchange::Account	0x02bd0aa0
[-] Exchange::IAccount			0x02bd0aa0
[-] m_pExchange	0x00C8FB44	Exchange::IExchange'	0x02bd0aa4
[-] m_acctName	"Its not broken Pty Ltd"	ATL::CStringT<wchar	0x02bd0aa8
[-] m_acctBalance	0x98a877 (10004599)	int	0x02bd0aac
[-] m_acctID	0x1 (1)	unsigned int	0x02bd0ab0
[-] sid	0x2 (2)	unsigned int	0x00c8f6e0
[-] amount	0x6a (106)	unsigned int	0x00c8f6e4
[-] debitPurchaseCost	0xffffec2 (-318)	int	0x00c8f6e8

Menu Contexte

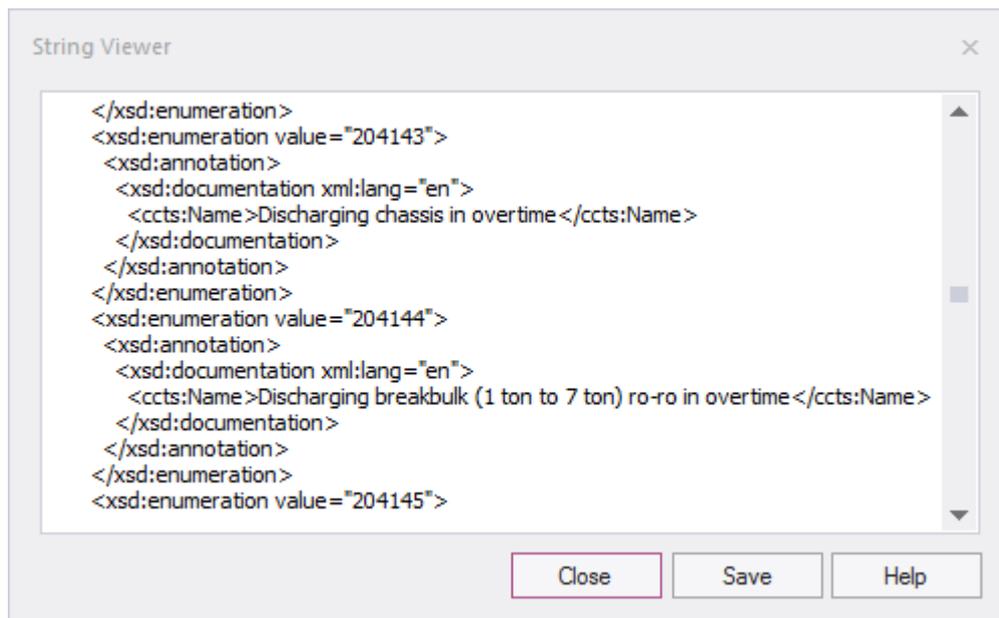
Facilité	Détail
Interrompre lorsque la variable est modifiée	Définissez des points d'arrêt de données sur la variable de mémoire sélectionnée pour arrêter l'exécution du débogueur à la ligne de code qui vient de provoquer la modification de la valeur de la variable.
Vue Mémoire à l'adresse	Affiche les valeurs brutes en mémoire à l'adresse sélectionnée, en hexadécimal et en ASCII.
Afficher dans la visionneuse String	Afficher la string variable dans la dialogue « Visionneuse String ».
Vider les membres de la variable dans un fichier	Capturez et stockez les variables sélectionnées dans un emplacement séparé ; un navigateur s'affiche pour sélectionner le nom de fichier .txt et le chemin d'accès au fichier appropriés.
Enregistrer Instantané de la variable	Capturez la valeur d'une variable à un moment précis de la vie de cette variable.
Comparer Instantanés variables	Comparer les valeurs d'une variable à différents moments de la vie de cette variable.
Copie	Copiez la variable sélectionnée dans le presse-papiers Enterprise Architect .
Ajouter State Exécuter de l'instance au Diagramme	Si vous avez ouvert un diagramme de modèle contenant un Object de la classe pour laquelle le code source est en cours de débogage, cette option met à jour cet Object avec l' State Exécuter représenté par la variable valeur .
Définir Point d'Arrêt conditionnel	Ajoutez un point d'arrêt à la position d'exécution actuelle avec une contrainte pour cette variable correspondant à sa valeur actuelle.

Voir le Contenu de Longues Chaînes

Pour des raisons d'efficacité, la fenêtre Variables locales n'affiche que des chaînes partielles. Cependant, vous pouvez afficher le contenu entier d'une variable string à l'aide du « Visionneuse String ».



Cet exemple montre la valeur d'une variable contenant le contenu d'un fichier de schéma XML.



Accéder

Depuis la fenêtre Éditeur de Code ou Locals	Cliquez-droit sur la variable string Afficher dans la visionneuse String
---	--

Variables Vue Déboguer dans Éditeurs de Code

Lorsqu'un point d'arrêt se produit, vous verrez toutes les variables locales dans cette fenêtre. Vous pouvez également inspecter les variables dans l'Éditeur de Code source en passant votre souris sur la référence. Voici quelques exemples.

```
public void Print()
{
    int n = 0;
    while(names[n].Length > 0)
    {
        names = {[4] names[0]=book, names[0]=book, names[1]=novel, names[2]=film}, ...}
        Document d = new Document(names[n++]);
        d.Print();
    }
}
```

```
public void Print()
{
    int n = 0;
    while(32-bit signed integer n=0 0)
    {
        Document d = new Document(names[n++]);
        d.Print();
    }
}
```

Note : la variable ne doit pas nécessairement être l'une des variables locales. Elle peut avoir une portée de fichier ou de module.

Variables Instantanés

Il est possible de prendre un « instantané » d'une variable lorsque votre programme atteint un point d'arrêt et d'utiliser cet instantané pour voir comment la valeur de la variable change à différents moments de sa vie. Le débogueur ne copie pas uniquement la valeur de la variable sélectionnée ; pour les variables complexes, il copie les valeurs de la variable sélectionnée et de chacune de ses hiérarchies de membres jusqu'à ce qu'il ne puisse plus trouver d'informations de débogage pour un membre ou qu'aucun autre membre ne puisse être trouvé.

Capture Variable Instantané

Étape	Action	
1	Dans l' Éditeur de Code , définissez deux points d'arrêt : un au début d'une fonction et un autre à la fin de la fonction.	
2	Au point d'arrêt de départ, cliquez-droit sur une variable dans la fenêtre Locales et sélectionnez l'option de menu 'Enregistrer la variable Instantané '.	
3	Exécuter l'application.	
4	Lorsque le point d'arrêt final est atteint, cliquez-droit sur la variable dans la fenêtre Locales et sélectionnez l'option 'Comparer les variables Instantanés '.	Une dialogue s'affiche, indiquant la valeur d'origine du premier instantané et la valeur actuelle du deuxième instantané comme illustré dans ce diagramme tiré du modèle EA.Exemple.

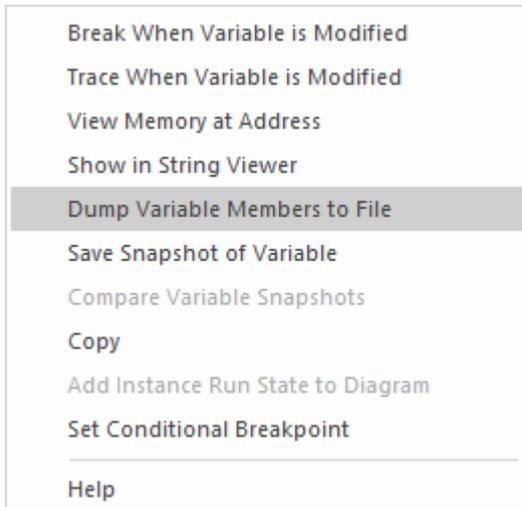
The screenshot shows a dialog box titled "CTrain* this" with a close button (X) in the top right corner. It contains a table with the following data:

Name	Address	Value 1	Value 2
int::Passengers	0x0003BEE4	0xb1 (177)	0xd6 (214)
int::Delay	0x0003BEF4	0x3c (60)	0x28 (40)

At the bottom of the dialog box, there are three buttons: "Close", "Save", and "Help".

Enregistrer la variable Instantané dans un fichier

Vous pouvez enregistrer l'état d'une variable dans un fichier en utilisant son menu contextuel cliquez-droit .



Ceci est un extrait du contenu du fichier.

```
73 00000006|0x00731F00|name|TObjectType::Type |value|TypeIsStation|
74 00000005|0x00731F08|name|wchar::Name |value|"Treasury"|
75 00000005|0x00731F0C|name|unsigned::Location |value|0x40 (64)|
76 00000003|0x0003BED8|name|float::Distance |value|0|
77 00000003|0x0003BEE0|name|int::Capacity |value|0x1f4 (500)|
78 00000003|0x0003BEE4|name|int::Passengers |value|0xd6 (214)|
79 00000003|0x0003BEE8|name|unsigned::Number |value|0x3 (3)|
80 00000003|0x0003BEF0|name|unsigned::Location |value|0x0 (0)|
81 00000003|0x0003BEF4|name|int::Delay |value|0x28 (40)|
```

Points d'Action

Points d'Action sont des points d'arrêt qui peuvent effectuer des actions. Lorsqu'un point d'arrêt est atteint, le script Point d'Action est invoqué par le débogueur et le processus continue à exécuter. Points d'Action sont des outils de débogage sophistiqués et fournissent aux développeurs experts une suite de commandes supplémentaire. Avec eux, un développeur peut modifier le comportement d'une fonction, capturer le point auquel un comportement change et modifier/détecter l'état d'un objet. Pour support ces fonctionnalités, Points d'Action peuvent modifier la valeur des variables locales et membres primitives, peuvent définir leurs propres « variables définies par l'utilisateur » et modifier l'exécution du programme.

Variables définies par l'utilisateur dans Points d'Action et Points d'Arrêt

Variables définies par l'utilisateur (UDV) :

- Fournir les moyens de définir une primitive ou string UDV dans les instructions Point d'Action
- Peut être utilisé dans les instructions de condition de plusieurs marqueurs/points d'arrêt
- Peut être facilement vu dans la même fenêtre Variables locales
- Les valeurs finales de tous les UDV sont enregistrées à la fin du débogage.

Dans la syntaxe UDV, le nom UDV :

- Doit être précédé d'un caractère # (dièse)
- Est insensible à la casse

Déclarations Point d'Action

Les instructions Point d'Action peuvent contenir des commandes set, des commandes goto et des commandes jmp.

commande set

Ensembles de valeurs variables. Une instruction Point d'Action peut contenir plusieurs commandes « set », qui doivent toutes précéder toute commande « goto ».

La syntaxe de la commande « set » est :

définir LHS = RHS

Où:

- **LHS** = le nom de la variable en tant que :
 - variable définie par l'utilisateur (UDV) telle que #myval
 - variable locale ou membre telle que strName ou this.m_strName
- **RHS** = la valeur à attribuer :
 - En tant que variable littérale ou locale
 - S'il s'agit d'un littéral, sous la forme de : integer, booléen, virgule flottante, caractère ou string

Commande set - Exemples de variables

Exemples d'UDV	Exemples de variables locales

définir #mychar = 'a'	définir ceci.m_nCount=0
définir #mystr = "une string "	définir bSuccess=false
définir #myint = 10	
définir #myfloat = 0,5	
définir #mytrue = vrai	

commande goto

Cette commande bascule l'exécution vers un numéro de ligne différent dans une fonction. Une instruction Point d'Action ne peut contenir qu'une seule commande goto, comme commande finale de l'instruction.

La syntaxe de la commande goto est :

aller à L

Où **L** est un numéro de ligne dans la fonction actuelle.

La commande **goto** utilise des points d'arrêt pour atteindre son objectif, ce qui provoque un léger retard dans l'exécution du code. Cela peut être perceptible dans les zones de code qui sont exécutées très fréquemment, vous préférerez donc peut-être utiliser la commande **jmp** dans ce type de code, pour obtenir le même détournement d'exécution mais avec moins de retard.

commande jmp

La commande **jmp** est effectivement la même que la commande **goto** .

jmp 125

aller à 125

Ces deux commandes provoquent le changement de l'exécution à la ligne 125.

Cependant, l'instruction **jmp** utilise en interne l'instrumentation pour diriger le programme vers le déplacement de l'exécution, tandis que l'instruction **goto** utilise des points d'arrêt pour ce faire, ce qui provoque un retard dans le traitement. La différence réside donc dans les performances supérieures de l'instruction **jmp** , en particulier lorsque des régions de code sont exécutées très fréquemment.

Opérateurs Integer

Lorsqu'une variable définie par l'utilisateur (UDV) existe et qu'elle est de type int , elle peut être incrémentée et décrétementée à l'aide des opérateurs ++ et --. Par exemple :

1. Créez un UDV et définissez sa valeur et son type sur une variable integer locale.
AP1 : définir #myint = nTotalSoFar
2. Augmenter l'UDV.
AP2 : #monint++
3. Diminuer l'UDV.
AP3 : #myint--

Opérations de minuterie

Points d'Action peuvent indiquer le temps écoulé entre deux points. Il n'y a qu'un seul minuteur disponible, qui est réinitialisé ou démarré avec la commande `startTimer`. Le temps écoulé actuel peut ensuite être imprimé avec la commande `printTimer`. Enfin, le temps total écoulé est imprimé et le minuteur est arrêté avec la commande `endTimer`.

Exemple de conditions Point d'Action

Avec des littéraux et des constantes :

- `(#monchar='a')`
- `(#mystr <> " ")`
- `(#monint > 10)`
- `(#monfloat > 0.0)`

Avec des variables locales :

- `(#myval == this.m_strValue)`
- `(#myint <> ceci->m_nCount)`
- `(#myint != ceci->m_nCount)`

Instruction Enregistrement

L'enregistrement des instructions peut être utile pour détecter les changements dans un comportement connu ; le point d'exécution (B) qui diverge d'une ou plusieurs exécutions précédentes (A). Les commandes sont :

- `recStart` - démarre l'enregistrement ou commence la comparaison si un enregistrement précédent existe
- `recStop` - termine l'enregistrement
- `recPause` - mettre en pause l'enregistrement
- `recResume` - reprend l'enregistrement

La commande **recStart** commence à enregistrer les instructions. Les instructions exécutées sont ensuite stockées. Lorsqu'une commande **recStop** est rencontrée, l'enregistrement est sauvegardé. Il ne peut y avoir qu'un seul enregistrement sauvegardé à la fois entre deux Points d'Action . Lorsqu'une **commande recStart** est rencontrée et qu'un enregistrement précédent existe, le débogueur commence à comparer chaque instruction suivante avec son enregistrement. Il peut effectuer de nombreuses comparaisons. Si et quand une différence est détectée, le débogueur s'arrête et la ligne de code où le comportement a changé s'affiche dans l'éditeur de code. L'itération de la comparaison est également imprimée.

L'enregistrement est stocké en mémoire par défaut, mais il peut également être stocké dans un fichier avec la syntaxe de commande :

Fichiers de démarrage `recspec`

Par exemple:

```
recStart c:\mylogs\onclickbutton.dat
```

Lorsqu'une commande **recStart** est rencontrée qui spécifie un fichier, et que ce fichier existe, il est chargé en mémoire et le débogueur entre immédiatement en mode de comparaison.

Expressions

Il n'y a pas de priorité implicite dans les expressions conditionnelles Point d'Arrêt , Point d'Action et Testpoint . Dans les expressions complexes, l'utilisation de parenthèses est obligatoire. Voir ces exemples :

Type	Exemple
Exemple de Point d'Action UDV	(#myint=1) AND (#mystr="Allemagne")
Exemples de variables locales	(this.m_nCount > 10) OR (nCount%1) (this.m_nCount > 10) OR (bForce)
Opérateurs d'égalité dans les expressions conditionnelles	<> - Pas égal != - Pas égal == - Égal = - Égal
Opérateur d'affectation dans Point d'Action	= - Affecte RHS à LHS
Opérateurs Arithmétique dans les expressions conditionnelles	/ - division + - plus - - moins * - multiplication % - module
Opérateurs logiques dans les expressions conditionnelles	AND - les deux doivent être vrais OR - l'un doit être vrai && - les deux doivent être vrais - il faut que ce soit vrai ^ - OR exclusif (un seul doit être vrai)

Voir Variables dans d'Autres Portées

Accéder

Ruban	Exécuter > Windows > Observateurs
Autre	Barre d'outils de la fenêtre Analyseur d'Exécution :  Observateurs

Vues

Vue	Description
Observateurs	<p>La fenêtre Observateurs est particulièrement utile pour le code natif (C, C++, VB) où elle peut être utilisée pour évaluer les éléments de données qui ne sont pas disponibles en tant que variables locales - éléments de données avec portée de module ou de fichier et éléments membres de classe statiques.</p> <p>Vous pouvez également utiliser la fenêtre pour évaluer les éléments membres de classe statiques en Java et .NET</p> <p>Pour ajouter une surveillance, tapez le nom de la variable à surveiller dans la barre d'outils et appuyez sur la touche Entrée.</p> <p>Pour examiner une variable membre de classe statique en C++, Java ou Microsoft .NET , entrez son nom complet :</p> <p><code>CMyClass::MyStaticVar</code></p> <p>Pour examiner un symbole de données C++ avec une portée de module ou de fichier, entrez simplement son nom.</p> <p>Les variables sont évaluées en regardant la portée actuelle, c'est-à-dire le module de la pile d'appel actuelle (vous pouvez modifier la portée à un point d'arrêt en double-cliquant sur la trame dans la Pile d'Appel).</p> <p>Si la variable globale existe dans un module différent, vous pouvez examiner la variable en préfixant le nom du module à la variable</p> <p><code>nom_module!nom_variable</code></p> <p>Il est assez facile de mal saisir les noms des éléments de données. Par conséquent, si vous faites ou trouvez une erreur, mettez en surbrillance la string , appuyez sur F2 et retapez le texte. Cela accélère également la résolution des éléments nommés dans le débogueur, en interrompant une recherche lorsque des éléments correspondants sont découverts.</p>
Histoire	<p>L'historique des éléments saisis est conservé. Les noms ou expressions saisis précédemment peuvent être sélectionnés à nouveau à l'aide des touches fléchées Haut et Bas dans la zone de texte de la barre d'outils. L'historique sera également conservé pour l'utilisateur sur n'importe quelle instance d' Enterprise Architect ou modèle sur la même machine.</p>

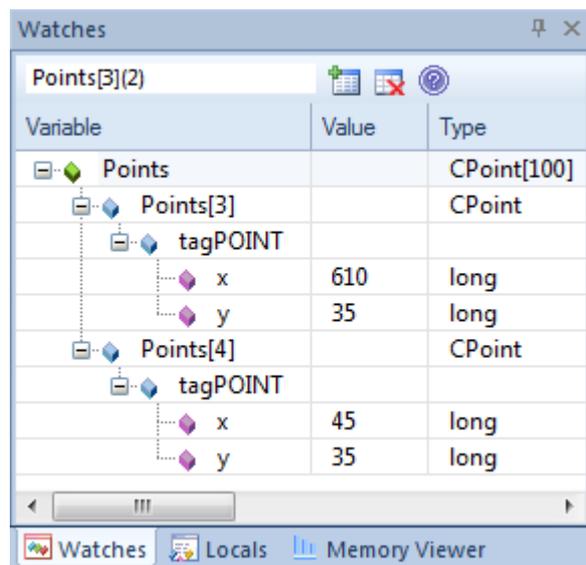
Voir Éléments du Réseau

Vous pouvez utiliser la fenêtre Observateurs pour inspecter un ou plusieurs éléments spécifiques d'un tableau.

Dans le champ à gauche de la barre d'outils de la fenêtre Observateurs, saisissez le nom de la variable du tableau suivi de l'élément de départ et du nombre d'éléments à afficher. L'élément de départ est placé entre crochets et le nombre d'éléments est placé entre parenthèses, c'est-à-dire :

variable[élément_de_début](nombre_d'éléments)

Par exemple, Points[3](2) affiche les quatrième et cinquième éléments du tableau Points, comme illustré.



Si vous avez entré Points[3], la fenêtre Observateurs affichera uniquement le troisième élément du tableau.

Accéder

Ruban	Exécuter > Windows > Observateurs
Autre	Barre d'outils de la fenêtre Analyseur d'Exécution :  Observateurs

Voir la Pile d'Appel

La fenêtre Pile d'Appel permet d'afficher tous les threads en cours d'exécution dans un processus. Elle peut être utilisée pour identifier quel thread est opérationnel, juste avant qu'une panne du programme ne se produise.

Lorsqu'une Simulation est active, la Pile d'Appel affiche le contexte d'exécution actuel de la simulation en cours. Cela comprend une pile de contexte distincte pour chaque « thread » de simulation simultanée.

Une trace de pile s'affiche à chaque fois qu'un thread est suspendu, par l'une des actions d'une étape ou par la rencontre d'un point d'arrêt. La fenêtre Pile d'Appel peut enregistrer un historique des modifications de la pile et vous permet de générer diagrammes Séquence à partir de cet historique.

Accéder

Ruban	Exécuter > Windows > Pile d'Appel
Autre	Barre d'outils de la fenêtre Analyseur d'Exécution :  Pile d'Appel

Utiliser pour

- Historique de la pile Vue pour comprendre l'exécution d'un processus
- Fils de discussion Vue
- Enregistrer une pile d'appels pour une utilisation ultérieure
- Enregistrer les modifications de la pile d'appels pour la génération diagramme Séquence
- Générer un diagramme Séquence à partir de la pile d'appels
- Vue la ligne de code associée dans le Source Éditeur de Code

Facilités

Facilité	Description
Indicateurs	<ul style="list-style-type: none"> • Une flèche rose met en évidence le cadre de pile actuel • Une flèche bleue indique un thread en cours d'exécution • Une flèche rouge indique un thread pour lequel un historique de trace de pile est en cours d'enregistrement
Enregistrer une Pile d'Appel dans un fichier .TXT	Pas disponible actuellement.
Enregistrer un fil de discussion dans une session Déboguer	<p>Pour enregistrer l'exécution d'un thread et diriger l'enregistrement vers la fenêtre Enregistrer et analyser, cliquez-droit sur le thread dans la Pile d'Appel et sélectionnez l'option de menu contextuel appropriée :</p> <ul style="list-style-type: none"> • « Enregistrer » : pour enregistrer manuellement le thread actuel pendant la

	<p>session de débogage</p> <p>Utilisé en conjonction avec les boutons « étape » du débogueur ; chaque fonction appelée en raison d'une commande d'étape est enregistrée dans la fenêtre Enregistrer et analyser</p> <ul style="list-style-type: none"> • « Enregistrement automatique » : pour effectuer un enregistrement automatique pendant une session de débogage <p>Lorsque vous sélectionnez cette icône, l'analyseur commence l'enregistrement et ne s'arrête pas tant que le programme n'est pas terminé, que vous n'arrêtez pas le débogueur ou que vous n'avez pas cliqué sur l'icône « Arrêter »</p>
Arrête d'Enregistrer	<p>Si vous avez démarré un enregistrement manuel ou automatique d'un fil de discussion, vous pouvez l'arrêter avant la fin ; sélectionnez le fil de discussion (indiqué par une flèche rouge) et soit :</p> <ul style="list-style-type: none"> • Cliquez sur le bouton  Arrête d'Enregistrer dans la barre d'outils ou • Cliquez-droit et sélectionnez l'option 'Stop'
Générer un Diagramme de Séquence à partir de la Pile d'Appel	<p>Pour générer diagramme Séquence à partir de la trace Pile d'Appel , soit :</p> <ul style="list-style-type: none"> • Cliquez sur le bouton  (Générer Diagramme de Séquence of Stack), ou • Cliquez-droit et sélectionnez l'option ' Générer Diagramme de Séquence '
Copier la pile dans Historique d'Enregistrement	<p>Pour ajouter immédiatement les détails de la pile à la fenêtre Enregistrer et analyser (pour une génération ultérieure de diagrammes Séquence), procédez comme suit :</p> <ul style="list-style-type: none"> • Cliquez sur le bouton  , ou • Cliquez-droit et sélectionnez l'option « Copier la pile dans l'historique des enregistrements »
Basculer Profondeur de Pile	<p>Pour basculer entre l'affichage de la pile complète et l'affichage uniquement des trames avec la source, cliquez sur le bouton  (Toggle Profondeur de Pile).</p>
Afficher le code associé dans Source Éditeur de Code	<p>Double-cliquez sur un thread/frame pour afficher la ligne de code concernée dans l' Éditeur de Code Source ; les variables locales sont également actualisées pour la frame sélectionnée.</p>

Créer Diagramme de Séquence de Pile d'Appel

La fenêtre Pile d'Appel enregistre un historique des modifications de la pile à partir duquel vous pouvez générer diagrammes Séquence .

Accéder

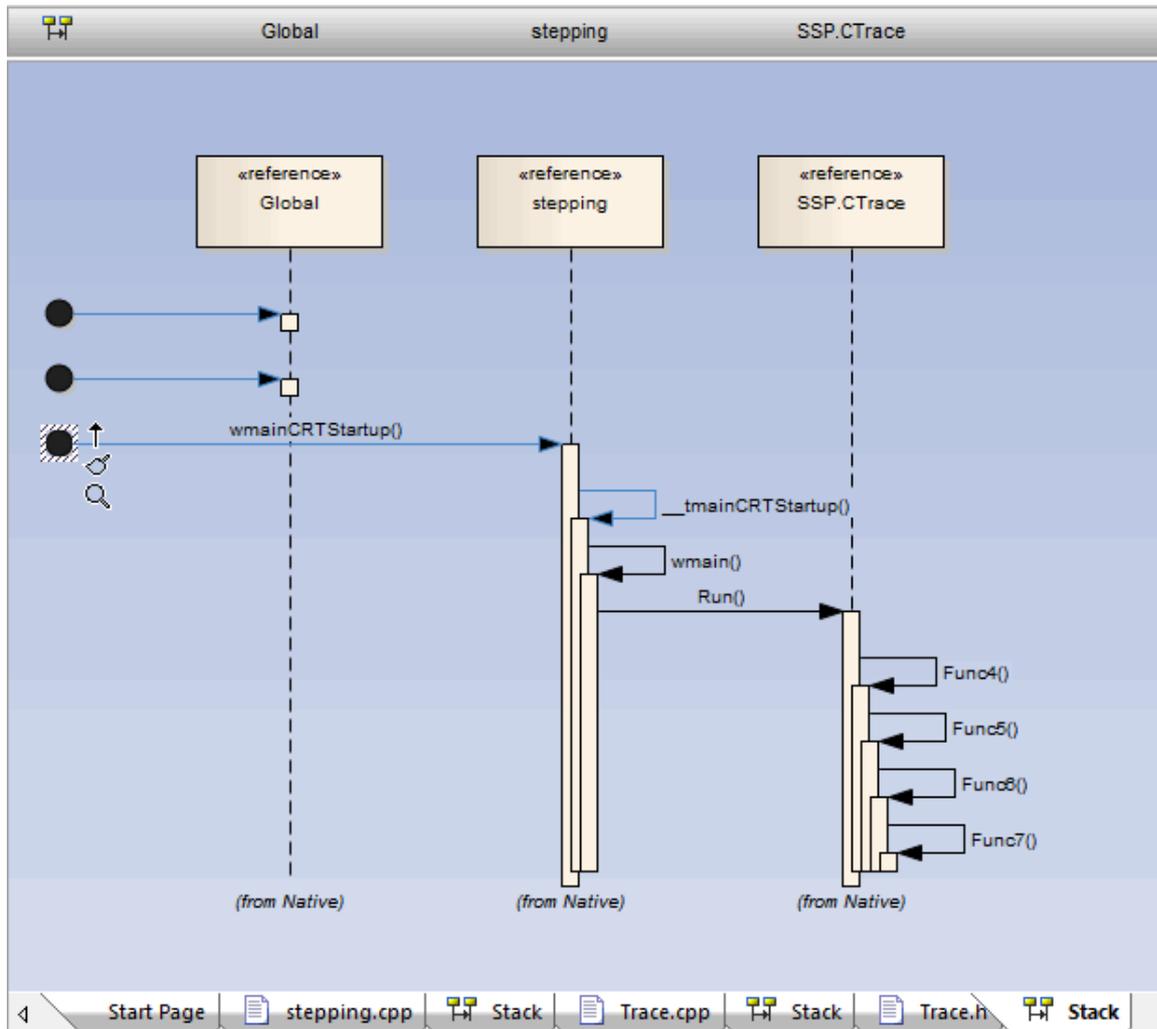
Ruban	Exécuter > Windows > Pile d'Appel
Autre	Barre d'outils de la fenêtre Analyseur d'Exécution :  Pile d'Appel

Utiliser pour

- Enregistrer les modifications Pile d'Appel pour la génération diagramme Séquence
- Générer un diagramme Séquence à partir de la Pile d'Appel

Pour générer un diagramme Séquence à partir de la Stack actuelle, cliquez sur le bouton  (Générer Diagramme de Séquence of Stack) de la barre d'outils de la fenêtre Pile d'Appel .

Cela génère immédiatement un diagramme Séquence dans le Diagramme Vue .



Inspecter Mémoire de Processus

En utilisant le visualiseur de mémoire, vous pouvez afficher les valeurs brutes de la mémoire en hexadécimal et en ASCII. Vous pouvez définir manuellement l'adresse mémoire dans le champ 'Adresse' (en haut à droite), ou cliquez-droit sur une variable dans la fenêtre des variables locales ou la fenêtre Observateurs et sélectionner l'option ' Vue mémoire à l'adresse'.

Accéder

Ruban	Exécuter > Windows > Visionneuse de mémoire
Autre	Barre d'outils de la fenêtre Analyseur d'Exécution :  Visionneuse de mémoire Depuis la fenêtre Locals ou Observateurs : Cliquez-droit sur une variable Mémoire Vue à l'adresse

Notes

- Le visualiseur de mémoire est disponible pour déboguer les applications Microsoft Native Code (C, C++, VB) exécutées sous Windows ou dans WINE sous Linux

Afficher Modules Chargés

Pour les applications .NET et Windows natives, vous pouvez répertorier les DLL chargées par le processus débogué à l'aide de la fenêtre Modules. Cette liste peut également inclure les fichiers symboliques associés (fichiers PDB) utilisés par le débogueur.

Accéder

Ruban	Exécuter > Windows > Modules
-------	------------------------------

Modules Vitrine

Colonne	Description
Chemin	Affiche le chemin du fichier du module chargé.
Adresse de chargement	Affiche l'adresse mémoire de base du module chargé.
Date de modification	Affiche la date du fichier local et l'heure à laquelle le module a été modifié.
Symboles Débogueur	Spectacles : <ul style="list-style-type: none">• Le type de symboles de débogage• Si les informations de débogage sont présentes dans le module, et• Si les informations de ligne sont présentes pour le module (requis pour le débogage)
Correspondance de fichiers de symboles	Indique la validité du fichier de symboles ; si la valeur est fausse, le fichier de symboles est obsolète.
Chemin du symbole	Affiche le chemin d'accès au fichier symbole, qui doit être présent pour que le débogage fonctionne.
Date de modification	Affiche la date et l'heure du fichier local auxquelles le fichier de symboles a été créé.

Traiter Exceptions à Première Chance

Accéder

Ruban	Exécuter > Outils > Débogueur > Traiter Exceptions à Première Chance
Autre	Barre d'outils de la fenêtre Débogueur :  Traiter Exceptions à Première Chance

Éléments de traitement

Élément	Description
Processus Débogueur	<p>Lorsqu'une application est en cours de débogage et que le débogueur est informé d'une exception, l'application est mise en pause et le débogueur répond de la manière pour laquelle il est configuré ; il peut :</p> <ul style="list-style-type: none"> Reprend l'application et laisse l'exception à l'application pour qu'elle la gère, ou Maintient l'application suspendue et transmet l'exception aux routines appropriées pour une résolution automatique ou une intervention manuelle
Exceptions de la deuxième chance	<p>Le débogueur Enterprise Architect adopte par défaut le premier comportement répertorié.</p> <p>Si l'application peut gérer l'exception, elle continue le traitement ; si elle ne peut pas gérer l'exception, le débogueur est à nouveau notifié et cette fois, il doit suspendre l'application et résoudre la condition d'exception.</p> <p>Dans ce comportement, étant donné que le débogueur a rencontré l'exception deux fois, on parle d'exception de seconde chance ; dans ce cas, si l'exception n'arrête pas l'exécution, elle est ignorée et vous évitez de passer du temps sur des conditions qui n'ont pas d'impact sur le résultat global du traitement.</p> <p>Vous pouvez travailler de cette manière sur des systèmes volumineux ou complexes qui impliquent invariablement des conditions d'exception quelque part dans les chemins de traitement.</p>
Exceptions de première chance	<p>Cependant, si vous souhaitez examiner chaque exception qui se produit dès qu'elle se produit, vous pouvez configurer le débogueur pour qu'il adopte le deuxième comportement.</p> <p>Étant donné que le débogueur répond à l'exception au premier contact, on parle d'exception de première chance.</p> <p>Vous pouvez travailler de cette manière avec des fonctions ou des routines individuelles qui doivent fonctionner correctement ou pas du tout.</p>
Sélection	<p>Sélectionnez l'option ' Traiter Exceptions à Première Chance ' pour déboguer les exceptions au premier contact.</p> <p>Désélectionnez l'option permettant de traiter les exceptions uniquement si l'application échoue lorsqu'elles se produisent.</p>

Débogueur juste à temps

Vous pouvez enregistrer le débogueur Enterprise Architect comme débogueur juste-à-temps du système d'exploitation, à appeler lorsqu'une application exécutée en dehors Enterprise Architect sur le système rencontre une exception ou se bloque. Dans ce cas, un blocage de l'application entraîne l'ouverture Enterprise Architect et l'affichage de la source et de la raison du blocage.

Accéder

Ruban	Exécuter > Outils > Débogueur > Définir comme Débogueur JIT
-------	---

Services

Enterprise Architect fournit deux services pour faciliter l'exécution de scripts à distance et le débogage à distance. Les services prennent principalement support Enterprise Architect exécuté sur Linux pour permettre aux utilisateurs d'exécuter des scripts shell Linux natifs et de déboguer des programmes Linux. Le service Satellite supporte Scripts d'Analyseur tandis que le service Agent supporte le débogage.

Accéder

Ruban	Exécuter > Outils > Services
-------	------------------------------

Le Service Satellite

Le service Satellite est responsable de l'exécution Scripts d'Analyseur sur la machine sur laquelle il s'exécute. La fonctionnalité peut aider les utilisateurs Linux à exécuter des programmes Linux natifs et des commandes shell directement, en contournant Wine. Le service peut être géré à partir du ruban, et il peut également être exécuter indépendamment d'un terminal.

Le shell Linux

Le shell par défaut utilisé par Enterprise Architect est « bash ». Pour remplacer le shell Linux utilisé par Enterprise Architect, ouvrez un terminal Linux, exécuter « wine regedit » et ajoutez une valeur string à cette clé de registre :

HKEY_CURRENT_USER\Software\Sparx Systems\EA400\EA\Options

où:

- nom de la clé : « LINUX »
- clé valeur : *chemin*

et *path* est le chemin Linux vers le programme shell " /bin/bash », par exemple.

Autorisations

Sous Linux, vous devez vérifier que les programmes de service disposent des autorisations appropriées. Les programmes se trouvent dans le dossier d'installation Enterprise Architect, dans le sous-répertoire « VEA/x86/linux ». Vérifiez que chacun des programmes de ce répertoire dispose des autorisations d'exécution définies pour le propriétaire.

Notes

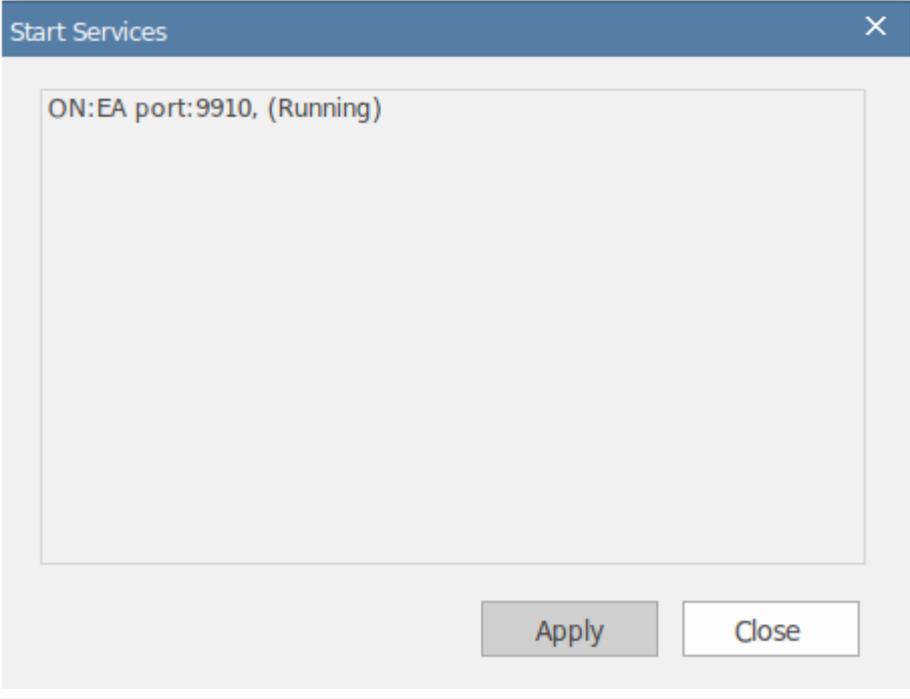
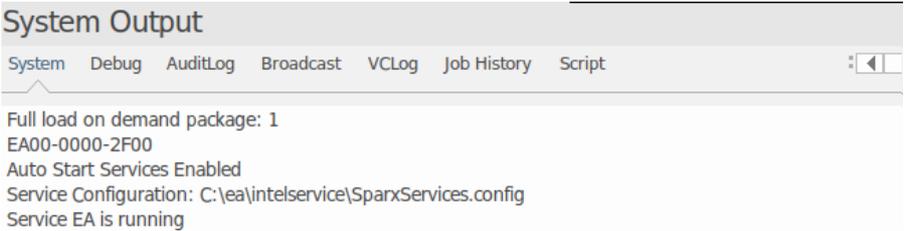
- Les services satellite sont activés dans les éditions Unified et Ultimate d' Enterprise Architect

Le Service d'Agent

Le service Agent est responsable de la gestion des sessions de débogage pour le débogueur GDB d' Enterprise Architect . Le service permet aux utilisateurs Enterprise Architect de déboguer des programmes Linux. Le service peut être géré à partir du ruban. Il peut également être exécuter indépendamment d'un terminal.

Le menu des services

Option	Description
Vue l'état de tous les services	Affiche une Vue qui répertorie l'état de chaque service Enterprise Architect nommé dans le fichier de configuration et son état.
Service Satellite	
Démarrer	Démarre le service. Le service écoute sur le port satellite configuré dans n'importe quelle page de services de script Analyzer.
Arrêt	Arrête le service.
Test	Teste l'état du Service Satellite , s'il est en cours d'exécution ou non.
Service d'Agent	
Démarrer	Démarre le service. Le service écoute sur le port d'agent configuré dans une page de services de script Analyzer.
Arrêt	Arrête le service.
Test	Teste l'état du Service d'Agent , qu'il soit en cours d'exécution ou non.
Service Code Miner	
Démarrer	<p>Cette option lit le fichier de configuration de service actuel et démarre les services configurés pour exécuter , et arrête l'exécution des services qui ne sont pas configurés pour exécuter . Un service est configuré si :</p> <ol style="list-style-type: none"> 1. Il est nommé dans le fichier de configuration. 2. Il a l'attribut <code>status:ON</code>.

	
Arrêtez tout	Cette option arrête tous les services en cours d'exécution.
Modifier le fichier de configuration	<p>Cette option promps le fichier de configuration de service à utiliser, puis ouvre ce fichier dans un éditeur de texte Enterprise Architect . Le système se souvient de l'emplacement du fichier.</p> <pre data-bbox="520 1093 1430 1621"> 31 # <number> - digits 32 # ----- 33 # 34 { 35 name=project1, 36 status=ON, 37 lazyload=true, 38 port=9910, 39 allow=localhost, 40 network=local, 41 autoupdate=true, 42 show=true, 43 logoutput=c:\My Documents\project1.txt, 44 loglevel=information warning error, 45 database=c:\My Documents\project1\project1.cdb 46 } 47 </pre>
Démarrer automatique avec EA	<p>Cette option démarre automatiquement les services ayant l'attribut « status:ON » lorsque le modèle s'ouvre.</p>  <p>Les messages enregistrés dans la fenêtre de sortie système ici lorsque le modèle est</p>

	ouvert indiquent que le service était déjà en cours d'exécution.
Arrêt automatique à la fermeture	Cette option arrête automatiquement l'exécution des services lorsque Enterprise Architect est fermé.

Fenêtre des services de l'analyseur

La fenêtre Analyzer Services affiche l'état de chacun des éléments suivants :

- Les services gérés par le script Analyzer actif (maintenu via l'option de ruban « Exécuter > Outils > Analyseur »)
- Tous les services locaux gérés séparément à l'aide du fichier de configuration de service (disponible à partir de l'option de ruban « Exécuter > Outils > Services > Service Code Miner > Modifier le fichier de configuration »)

Type	O/S	Name	Source	Status
Sparx	Linux	Linux Satellite Service Native Linux Shell Service for EA WINE installations localhost:9900	Analyzer RNO 160 - x64	Running
Sparx	Windows	Windows Satellite Service Native Windows Service for building Visual Studio Projects 192.168.20.7:9000	Analyzer RNO 160 - x64	Running
Remote		Intel Service Intel Service for Code Editors and Code Analyzer 172.16.18.16:9910	Analyzer RNO 160 - x64	Running
Personal	Linux	EA160 c:\codeminer\ea160\ea.cdb localhost:9100	Config Config	Stopped

La fenêtre vous permet de voir en un coup d'œil quel service Intel vous utilisez, par exemple, ou que le service Windows que vous utilisez pour travailler dans Enterprise Architect est en cours d'exécution.

Toutes les données de la fenêtre sont en lecture seule, mais vous pouvez modifier le service en cliquant sur le bouton Modifier. Cela affiche la page « Options privées - Services » de la fenêtre de l'éditeur Scripts d'Analyseur, que vous pouvez mettre à jour selon vos besoins.

Accéder

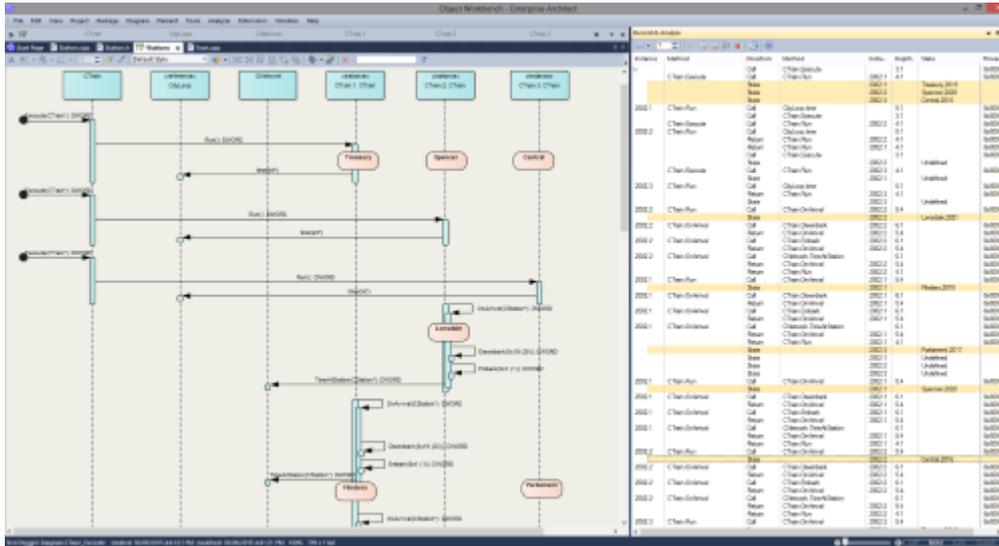
Ruban	Exécuter > Outils > Services > Vue l'état de tous les services Démarrer > Toutes Windows > Exécuter > Analyser Analyser Services
Raccourcis Clavier	Alt+4 Analyser Services d'analyse

Champs de la fenêtre des services d'analyse

Champ	Description
Ordinateur	Le poste de travail sur lequel les services s'exécutent.

Système	Le système d'exploitation sous lequel fonctionne le poste de travail.
EA	Le numéro de version et le numéro de build de la version d' Enterprise Architect que vous utilisez.
Modèle	Le nom du modèle sur lequel vous travaillez actuellement.
Type	Le type de service. Les services répertoriés comme type « Sparx » sont des services Enterprise Architect .
Système d'exploitation	Le système d'exploitation sous lequel le service s'exécute.
Nom	<p>Le nom et la description du service.</p> <ul style="list-style-type: none">• Le Service Satellite Linux est géré automatiquement par Enterprise Architect lorsque l'application s'exécute sur Linux sous WINE• Le Service Satellite Windows est un service optionnel généralement utilisé dans les installations WINE pour aider à créer des projets Visual Studio sur une machine Windows distante telle qu'une machine virtuelle (Virtual Machine) ; le service ne s'exécute que sur Windows et est géré localement à l'aide des services Windows sur la machine elle-même
Source	Le groupe de scripts ou le fichier de configuration définissant le service.
Statut	<p>L'état du service.</p> <p>Lorsque vous ouvrez la fenêtre pour la première fois, chaque service a initialement le statut « Test » pendant que le système évalue le service. Le statut change ensuite pour la valeur appropriée, comme « En cours d'exécution » ou « Arrêté ».</p>
Modifier	Cliquez sur ce bouton pour afficher ou modifier la définition du service.

Enregistrement



diagrammes Séquence sont une aide précieuse pour comprendre le comportement. diagrammes de collaboration de classe peuvent également être utiles. En plus de ceux-ci, un Graphique d'Appel est parfois exactement ce dont nous avons besoin. Mais encore une fois, si vous disposez de ces informations, vous pouvez les utiliser pour documenter un cas d'utilisation, et pourquoi ne pas créer un domaine Test pendant que vous y êtes ? Enterprise Architect Analyzer peut générer tout cela pour vous et à partir d'un seul enregistrement. Il le fait en enregistrant un programme en cours d'exécution et il fonctionne sur toutes les plateformes les plus populaires.

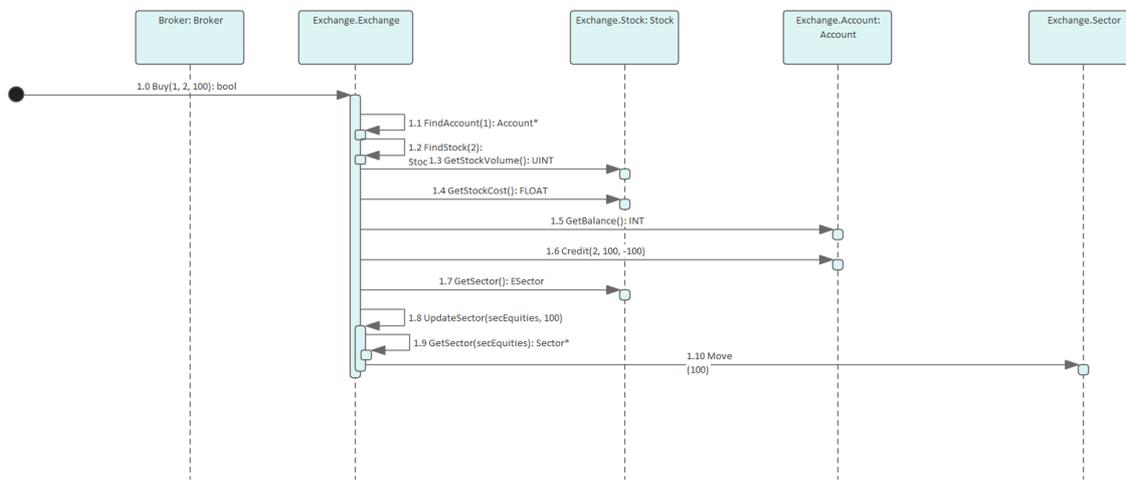
Accéder

Ruban

Exécuter > Outils > Enregistreur

Aperçu

Dans sa forme la plus simple, un diagramme Séquence peut être produit en quelques étapes, même à partir d'un tout nouveau modèle. Vous n'avez même pas besoin de configurer un script Analyzer. Ouvrez l'éditeur de code Enterprise Architect (Ctrl+Maj+O), placez un marqueur d'enregistrement dans une fonction de votre choix, puis attachez le Débugueur Enterprise Architect à un programme exécutant ce code. Chaque fois que cette fonction est appelée, son comportement sera capturé pour former un historique d'enregistrement. À partir de cet historique, ces diagrammes peuvent être facilement créés.

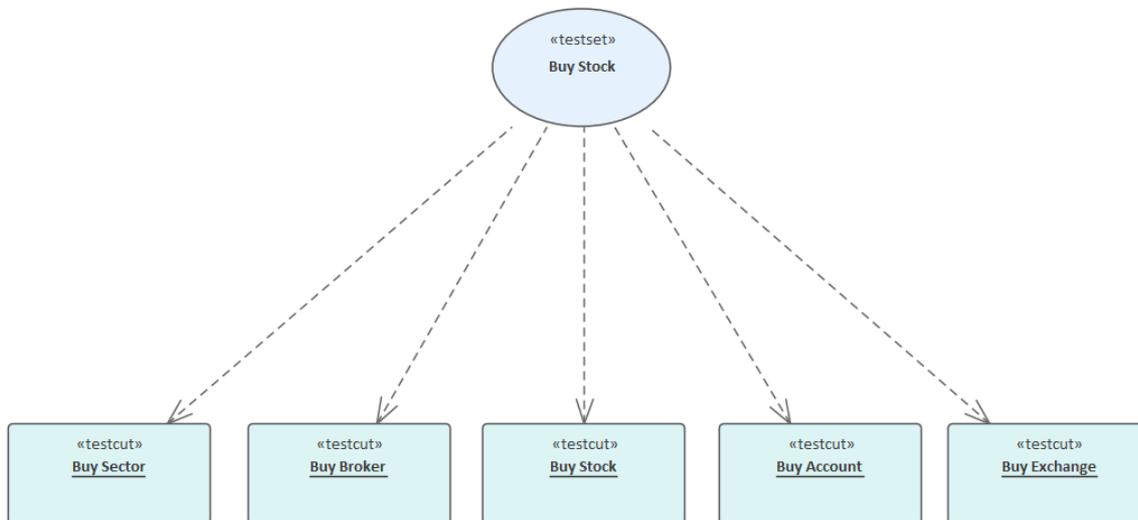


Sequence diagram generated in Enterprise Architect using recording marker in a Use Case

Le diagramme Séquence de l'enregistrement Exemple Modèle .



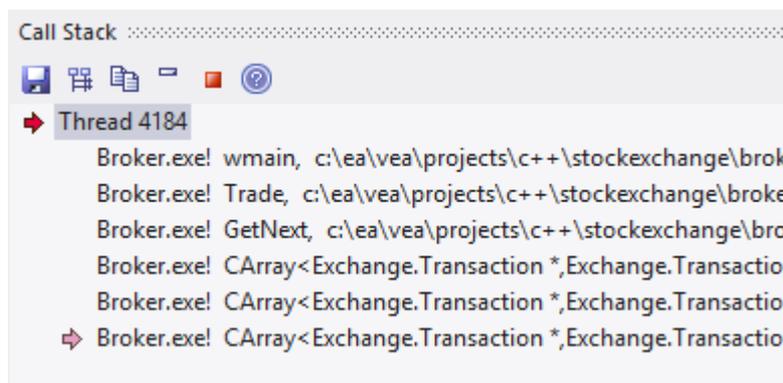
Le diagramme de collaboration de classe du même enregistrement.



Le diagramme du domaine Test du même enregistrement.

Bien sûr, un Analyzer Script reste la meilleure idée et ouvre un environnement de développement incroyablement riche, mais il convient de noter que des résultats significatifs peuvent être obtenus sans en avoir un. Cela est également vrai pour les outils Enterprise Architect Débugueur et Profiler.

Un point intéressant : vous pouvez visualiser le comportement d'un thread pendant son enregistrement. L'affichage de la Pile d'Appel pendant un enregistrement affichera les mises à jour de la pile d'un thread en temps réel, un peu comme une animation. C'est un bon outil de retour d'information et dans certaines circonstances, cela peut suffire.



Aperçu des Fonctionnalités

Génération Diagramme

- diagramme Séquence
- diagramme de collaboration de classe
- diagramme du domaine Test
- Capture de transition State
- Graphique d'Appel

Contrôle

- Support des modèles multithread et monothread
- Support du contrôle de la profondeur de la pile
- Support des filtres pour restreindre la capture
- support des caractères génériques de filtrage

- Mise à jour de la pile en temps réel

Intégration

- Modèle de classe
- Domaine Test
- Statemachine
- Statemachines Exécutables
- Tests unitaires

Plateformes

- Microsoft .NET
- Microsoft natif
- Java
- PHP
- GDB
- Androïde

Exigences

- Enregistrement est disponible pour les utilisateurs de toutes les éditions d' Enterprise Architect

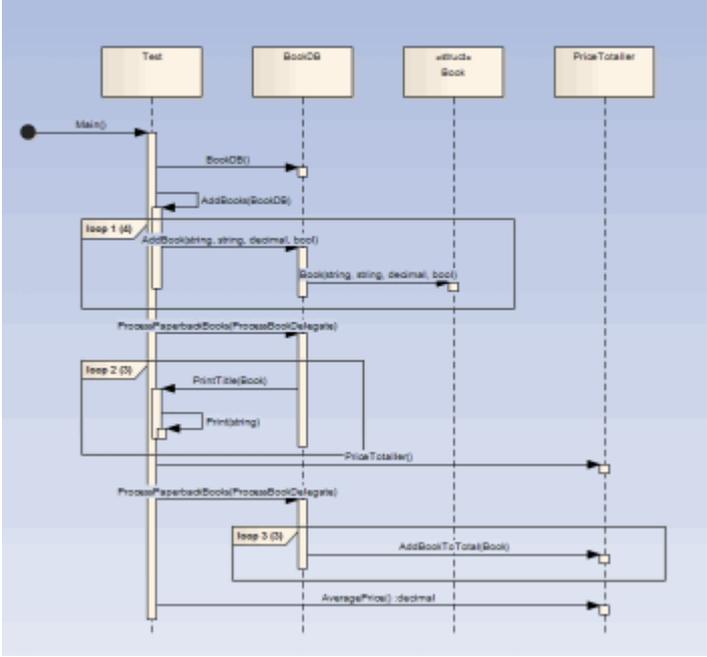
Notes

- Les fonctionnalités de débogage et d'enregistrement de l' Analyseur d'Exécution Visuelle ne sont pas supportées pour la plateforme serveur Java 'Weblogic' d'Oracle

Comment ça marche

Cette rubrique explique comment l'Analyseur d'Exécution Visuelle génère des diagrammes Séquence.

Explication

Points	Détail
Usage	<p>L'Analyseur d'Exécution Visuelle permet de générer un diagramme Séquence à partir d'enregistrements de l'exécution en direct d'une application. Au fur et à mesure de l'exécution de l'application, l'historique de chaque thread est enregistré. Cet historique peut être utilisé pour générer le diagramme Séquence.</p> <p>Il s'agit d'un diagramme Séquence généré à partir d'un programme qui calcule le prix des livres :</p>  <pre> sequenceDiagram participant Test participant BookDB participant Methode Book participant PriceTotaler Test->>Main() Test->>BookDB: BookDB() BookDB->>Test: AddBooks(BookDB) loop 1 (4) Test->>Methode Book: AddBook(string, string, decimal, bool) Methode Book->>BookDB: Book(string, string, decimal, bool) end Test->>Methode Book: ProcessFeedbackBooks(ProcessBookDelegate) loop 2 (3) Test->>PriceTotaler: PrintTitle(Book) PriceTotaler->>Methode Book: Print(string) end Test->>Methode Book: ProcessFeedbackBooks(ProcessBookDelegate) loop 3 (3) Test->>PriceTotaler: AddBookToTotal(Book) PriceTotaler->>Methode Book: AveragePrice() : decimal end </pre> <p>Comment l'enregistreur sait-il ce qu'il doit enregistrer ?</p> <ul style="list-style-type: none"> • L'enregistreur fonctionne à partir de marqueurs d'enregistrement ; ceux-ci sont placés par vous dans les fonctions qui vous intéressent <p>Les piles d'appels en Java peuvent s'étendre au-delà de ce que l'œil peut voir. Comment pouvons-nous limiter l'enregistrement à seulement dix images ?</p> <ul style="list-style-type: none"> • L'enregistreur est contrôlé par la profondeur définie sur la barre d'outils de l'enregistreur ou associée à un ensemble Marqueur stocké dans le modèle
C'est la vraie chose	<p>Lors de l'enregistrement, l'application cible n'est pas modifiée ; aucune instrumentation d'image ou de module n'est effectuée. Un enregistrement produit à l'aide d'une version « Release » d'un programme est un document fiable sur ce qu'un programme a fait.</p>
Par où commencer	<p>Nous avons une application serveur très volumineuse. Par où commencer ? Si vous avez peu ou pas de compréhension du programme que vous souhaitez enregistrer et peu ou pas de modèle sur lequel vous appuyer, il est peut-être préférable de</p>

	<p>commencer par le Profiler. L'exécution du Profiler tout en utilisant un programme d'une manière spécifique peut rapidement identifier les cas d'utilisation à partir des points d'entrée et des graphiques d'appels présentés. Le fait de disposer de ces connaissances peut vous permettre de vous concentrer sur les domaines non couverts et d'enregistrer ces fonctions.</p> <p>Si vous disposez du code source, il vous suffit de placer un marqueur d'enregistrement dans une fonction qui vous intéresse. Nous vous déconseillons de placer plusieurs marqueurs d'enregistrement dans plusieurs fonctions en même temps. Dans la pratique, cela s'est avéré moins utile. Où placez-vous un marqueur d'enregistrement ? Pour les programmes UI Windows et par rapport à certains cas d'utilisation commerciale, vous pouvez commencer par en placer un dans les gestionnaires d'événements pour un message qui semble le plus pertinent. Si vous étudiez une fonction utilitaire, définissez simplement un marqueur d'enregistrement de méthode au début ou à proximité.</p> <p>Pour les services, les démons et les processus par lots, vous souhaitez peut-être profiler le programme une fois pour chaque comportement intéressant et utiliser le rapport pour explorer les zones non couvertes.</p>
Conseil	C'est une bonne idée de jeter un rapide coup d'œil à la fenêtre Points d'Arrêt et Marqueurs avant le débogage, et de vérifier que les marqueurs répertoriés ici correspondent à ce que vous attendez.
Scénarios	<ul style="list-style-type: none">• Microsoft C natif et C++, VB (Programmes Windows , services Windows, programmes de console, serveurs COM, modules IIS ISAPI, Legacy)• Microsoft .NET (ASP.NET, Windows Presentation Foundation (WPF), Windows Forms, Workflow Services, appareils, émulateurs)• Java (Applications, Applets, Servlets, Beans)• Androïde (en utilisant le pont de débogage Android pour les appareils et les émulateurs)• PHP (Scripts de sites Web)• GDB (Interopérabilité Windows / Linux)

L' Historique d'Enregistrement

Lorsque l'analyse d'exécution d'une application rencontre des marqueurs d'enregistrement définis par l'utilisateur, toutes les informations enregistrées sont conservées dans la fenêtre Enregistrer et analyser.

Accéder

Ruban	Exécuter > Outils > Enregistreur > Ouvrir Enregistreur
-------	--

Facilités

Facilité	Informations/Options
Affichage des informations	<p>Les colonnes de la fenêtre Enregistrer et analyser sont :</p> <ul style="list-style-type: none"> • Séquence - Le numéro de séquence unique • Threads - L' ID du thread du système d'exploitation • Delta - Le temps CPU écoulé depuis le début de la séquence • Méthode - Il y a deux colonnes Méthode : la première affiche l'appelant pour un appel ou pour une trame en cours en cas de retour ; la seconde affiche la fonction appelée ou la fonction à laquelle elle renvoie • Direction - Mouvement du cadre de pile, peut être un appel, un retour, State , Point d'Arrêt ou une évvasion (l'évasion est utilisée en interne lors de la production d'un diagramme Séquence , pour marquer la fin d'une itération) • Profondeur - La profondeur de la pile au moment d'un appel ; utilisée dans la génération de diagrammes Séquence • State - L'état entre les séquences • Source - Il y a deux colonnes Source : la première affiche le nom du fichier source et le numéro de ligne de l'appelant pour un appel ou, s'il s'agit d'un retour, pour une trame en cours ; la seconde affiche le nom du fichier source et le numéro de ligne de la fonction appelée ou de la fonction renvoyée • Instance - Il existe deux colonnes Instance, qui n'ont des valeurs que lorsque le diagramme Séquence produit contient des transitions State ; les valeurs se composent de deux éléments séparés par une virgule : le premier élément est un numéro unique pour l'instance de la classe qui a été capturée, et le second est l'instance réelle de la classe <p>Par exemple : supposons qu'une classe « CName » ait une valeur interne de 4 567 et que le programme ait créé deux instances de cette classe ; les valeurs pourraient être :</p> <ul style="list-style-type: none"> - 4567,1 - 4567,2 <p>La première entrée montre la première instance de la classe et la deuxième entrée montre la deuxième instance</p>
Opérations sur l'information	<p>La barre d'outils de la fenêtre Enregistrer et analyser fournit une gamme de facilités permettant de contrôler l'enregistrement de l'exécution d'un script Analyzer.</p> <p>Vous pouvez effectuer un certain nombre d'opérations sur les résultats d'un</p>

	enregistrement, en utilisant le menu contextuel de la fenêtre Enregistrer et analyser, une fois l'enregistrement terminé.
--	---

Notes

- La case à cocher en regard de chaque opération permet de contrôler si cet appel peut ou non être utilisé pour créer un diagramme Séquence , de classe de domaine Test ou de classe collaborative à partir de cet historique.
- En plus d'activer ou de désactiver l'appel à l'aide de la case à cocher, vous pouvez utiliser les options du menu contextuel pour activer ou désactiver un appel entier, tous les appels à une méthode donnée ou tous les appels à une classe donnée.

Diagramme Fonctionnalités

Lorsque vous générez un diagramme Séquence , il inclut ces fonctionnalités :

Fonctionnalités

Fonctionnalité	Détail
Références	<p>Lorsque l' Analyseur d'Exécution Visuelle ne peut pas faire correspondre un appel de fonction à une opération dans le modèle, il crée quand même la Séquence mais crée également une référence pour toute Classe qu'il ne peut pas localiser.</p> <p>Il le fait pour toutes les langues.</p>
Fragments	<p>Les fragments affichés dans le diagramme Séquence représentent des boucles ou des itérations d'une ou plusieurs sections de code.</p> <p>L' Analyseur d'Exécution Visuelle tente de faire correspondre la portée de la fonction avec les appels de méthode pour représenter aussi précisément que possible l'exécution visuellement.</p>
States	<p>Si une Statemachine a été utilisée pendant le processus d'enregistrement, toutes les transitions dans State sont présentées après l'appel de méthode qui a provoqué la transition.</p> <p>States sont calculés au retour de chaque méthode à son appelant.</p>

Configuration pour Enregistrement

Cette section explique comment préparer l'enregistrement de l'exécution de l'application.

Mesures

Étape
<p>Prérequis - Pour configurer l'environnement d'enregistrement diagrammes Séquence vous devez :</p> <ul style="list-style-type: none">• J'ai terminé la configuration de base pour Build & Déboguer et créé des scripts d'analyse d'exécution pour le Paquetage• Être capable de déboguer avec succès l'application
<p>Affinez la mise au point d'un enregistrement en appliquant des filtres.</p>
<p>Contrôlez les détails d'un enregistrement en ajustant la profondeur de la pile.</p>

Contrôle Profondeur de Pile

Lors de l'enregistrement de points de niveau particulièrement élevé dans une application, le nombre d'images de pile peut entraîner la collecte d'une grande quantité d'informations. Pour obtenir une image plus rapide et plus claire, il est préférable de limiter la profondeur de la pile sur la barre d'outils de l'un ou l'autre des éléments suivants :

- La vitrine Point d'Arrêt et Marqueurs ou
- La fenêtre Enregistrer et analyser

Accéder

Ruban	Exécuter > Outils > Enregistreur > Ouvrir Enregistreur
-------	--

Définir la profondeur de la pile d'enregistrement

Vous définissez la profondeur de la pile d'enregistrement dans le champ numérique de la barre d'outils de la fenêtre Points d'Arrêt & Marqueurs ou de la fenêtre Enregistrer & Analyser :



Par défaut, la profondeur de la pile est définie sur trois images. La profondeur maximale qui peut être saisie est de 30 images.

La profondeur est relative à la trame de pile où un marqueur d'enregistrement est rencontré ; ainsi, lorsque l'enregistrement commence, si la trame de pile est 6 et que la profondeur de pile est définie sur 3, le Débogueur enregistre les trames 6 à 8.

Pour les situations où la pile est très grande, il est recommandé d'utiliser d'abord une faible profondeur de pile de 2 ou 3. À partir de là, vous pouvez augmenter progressivement la profondeur d'enregistrement de la pile et insérer des marqueurs d'enregistrement supplémentaires pour agrandir l'image jusqu'à ce que toutes les informations nécessaires soient affichées.

Placer des Marqueurs d'Enregistrement

Cette section explique comment placer des marqueurs d'enregistrement, qui vous permettent d'enregistrer silencieusement l'exécution du code entre deux points. L'enregistrement peut être utilisé pour générer un diagramme Séquence .

Comme ce processus enregistre l'exécution de plusieurs threads, il peut être particulièrement utile pour capturer des séquences pilotées par des événements (telles que les événements de souris et de minuterie).

Accéder

Ruban	Exécuter > Windows > Points d'Arrêt
-------	-------------------------------------

Actes

Action
Différents marqueurs d'enregistrement peuvent être utilisés pour enregistrer le flux d'exécution ; consultez les liens associés pour obtenir des informations sur les propriétés et l'utilisation de ces marqueurs.
Gérez les points d'arrêt dans la fenêtre Point d'Arrêt & Marqueurs .
Activer et désactiver les marqueurs.
Travailler avec Ensembles de Marqueurs - lorsque vous créez un point d'arrêt ou un marqueur, il est automatiquement ajouté à un ensemble de marqueurs, soit l'ensemble par défaut, soit un ensemble que vous créez dans un but spécifique.

Notes

- *La Gestion des Point d'Arrêt et Marqueurs* le sujet (Ingénierie de Logiciel) décrit une perspective différente

Coffret Marqueurs d'Enregistrement

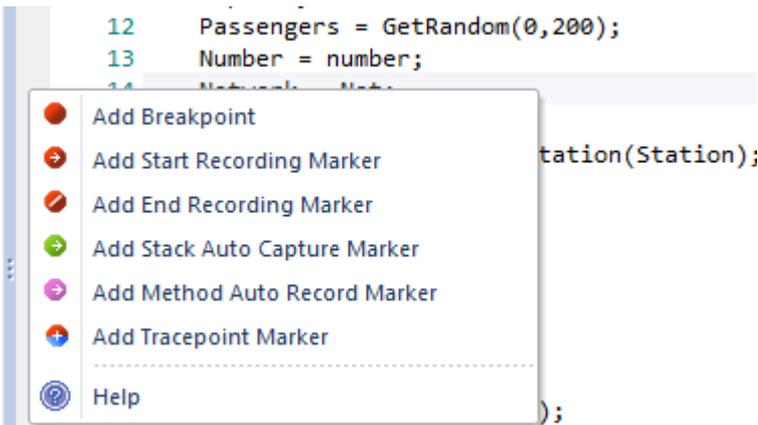
Marqueurs sont définis dans l'éditeur de code source. Ils sont placés sur une ligne de code ; lorsque cette ligne de code s'exécute, l'Analyseur d'Exécution effectue l'action d'enregistrement appropriée au marqueur.

Accéder

Utilisez l'une des méthodes décrites ici pour afficher la fenêtre Éditeur de Code et charger le code source associé à la Classe ou à l'élément Classe sélectionné.

Ruban	Exécuter > Source > Modifier > Modifier la source de l'élément Exécuter > Source > Modifier > Fichier Open Source
Raccourcis Clavier	Sur un élément appuyez sur Ctrl+E ou F12 Pour afficher le navigateur « Fichier Open Source », appuyez sur Ctrl+Alt+O.

Définir un marqueur d'enregistrement

Étape	Action
1	Ouvrez le code source à déboguer, dans l'éditeur de code source intégré.
2	<p>Recherchez la ligne de code appropriée et cliquez-droit dans la marge de gauche (Point d'Arrêt) pour faire apparaître le menu contextuel du point d'arrêt/marqueur ; sélectionnez le type de marqueur requis :</p> 
3	Si un Marqueur Démarrer Enregistrement a été défini, vous devez également définir un Marqueur Enregistrement de fin .

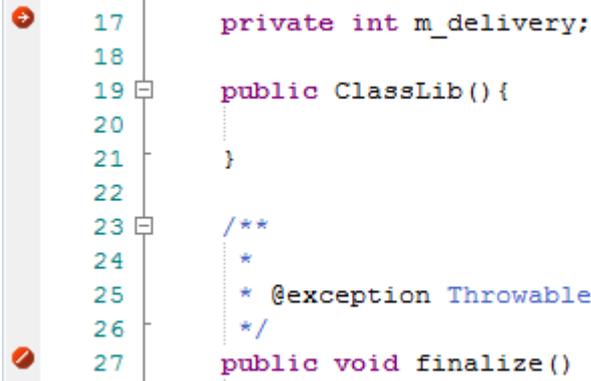
Types de Marqueurs

Marqueurs sont vraiment fantastiques. Inhabituels par leur encombrement très léger lorsqu'ils sont utilisés avec précaution, leur impact sur les performances des programmes enregistrés peut être négligeable. Marqueurs existent en plusieurs versions (en fait en plusieurs couleurs) et d'autres sont toujours ajoutés. Ils sont placés et sont visibles dans la marge gauche de l'éditeur, vous aurez donc besoin d'un code source.

Utiliser pour

- Enregistrer une seule fonction
- Enregistrer les parties d'une fonction
- Cas d'utilisation couvrant plusieurs fonctions
- Enregistrer les piles d'appels
- Générer diagrammes Séquence
- Générer diagrammes de domaines Test
- Générer diagrammes de collaboration de classes

Référence

Marqueur	Détail
Marqueurs de Démarrer / fin Enregistrement	<p>Placez les marqueurs aux lignes de début et de fin du code à enregistrer. Il n'est pas nécessaire que ces lignes se trouvent dans la même fonction.</p>  <pre> 17 private int m_delivery; 18 19 public ClassLib() { 20 21 } 22 23 /** 24 * 25 * @exception Throwable 26 */ 27 public void finalize() </pre> <p>Lorsque le programme rencontre un marqueur de début d'enregistrement, un nouvel enregistrement est lancé (<i>la caméra commence à tourner !</i>). Lorsqu'un marqueur de fin est rencontré, l'enregistrement en cours se termine (<i>c'est une prise</i>). La manière dont vous utilisez ces marqueurs dépend de vous et de vos connaissances du système dont vous avez la charge.</p> <p>Éléments avancés (marqueurs imbriqués) :</p> <p>Si un marqueur Démarrer l'enregistrement est rencontré alors qu'un enregistrement est en cours, mais que <i>la capture est inhibée par la valeur Stack depth in use</i>, un enregistrement séparé sera lancé. Chaque enregistrement est conservé sur une pile. Lorsqu'un enregistrement se termine, il est supprimé. Cette technique peut être utilisée dans Enterprise Architect pour enregistrer et restituer des scènes dans des systèmes très complexes. Elle ressemble à l'assemblage de courtes scènes d'une vidéo pour créer une bande-annonce. Si vous ne souhaitez enregistrer qu'une seule</p>

<p>Méthode marqueur d'Auto-Enregistrement</p>	<p>fonction, vous devez utiliser un marqueur d'enregistrement automatique.</p> <p>Une Méthode marqueur d'Auto-Enregistrement permet d'enregistrer une fonction particulière. Le débogueur arrêtera automatiquement l'enregistrement une fois la fonction terminée. C'est une bonne chose car l'enregistrement est une opération intensive.</p> <p>La fonction marqueur combine un marqueur Démarrer Enregistrement et un marqueur Fin Enregistrement en un seul, de sorte que l'enregistrement est exécuté après le point marqueur et s'arrête toujours lorsque cette fonction se termine.</p> <pre> 185 // 186 // CRecurrenceDlg message handlers 187 188 BOOL CRecurrenceDlg::OnInitDialog() 189 { 190 CBCGPDIALOG::OnInitDialog(); 191 192 UINT nMask = 193 CBCGPDIALOG::DTM_SPIN 194 CBCGPDIALOG::DTM_DATE 195 CBCGPDIALOG::DTM_TIME 196 CBCGPDIALOG::DTM_CHECKBOX 197 CBCGPDIALOG::DTM_DROPCALENDAR 198 CBCGPDIALOG::DTM_CHECKED; 199 200 UINT nFlags = CBCGPDIALOG::DTM_CHECKED CBCGPDIALOG::DI 201 //----- 202 // Setup date fields: </pre> <p>Les marqueurs Enregistrement peuvent être imbriqués. Lorsqu'un nouveau Méthode marqueur d'Auto-Enregistrement est atteint pendant l'enregistrement, la profondeur de la pile sur laquelle enregistrer sera étendue pour inclure la méthode actuelle et la profondeur requise à partir de cette fonction.</p>
<p>Piler marqueur de Auto-Capture</p>	<pre> 76 /* End - EA generated code for Parts and Ports */ 77 /* Begin - EA generated code for Activities and I 78 public void ClassLib_ActivityGraphWithActionPin() 79 { </pre> <p>Les marqueurs de pile vous permettent de capturer toutes les traces de pile uniques qui se produisent à un moment donné d'une application ; ils fournissent une image rapide et utile de l'endroit à partir duquel un point d'une application est appelé.</p> <p>Pour insérer un marqueur à l'endroit souhaité dans le code, cliquez-droit sur la ligne et sélectionnez l'option 'Ajouter Marqueur de capture automatique de pile'.</p> <p>Chaque fois que le débogueur rencontre le marqueur, il exécute une trace de pile ; si la trace de pile ne figure pas dans l'historique d'enregistrement, elle est copiée et l'application continue de s'exécuter.</p>
<p>Limiter la profondeur d'enregistrement</p>	<p>Vous pouvez limiter la profondeur des images dans n'importe quel enregistrement à l'aide du contrôle de profondeur de pile sur les barres d'outils de l'enregistreur et des points d'arrêt.</p>

La Fenêtres Points d'Arrêt et Marqueurs

A l'aide de la fenêtre Points d'Arrêt & Marqueurs , vous pouvez appliquer un contrôle à Analyse d'Exécution Visuelle lors de l'enregistrement de l'exécution pour générer diagrammes Séquence ; par exemple, vous pouvez :

- Activer, désactiver et supprimer les marqueurs
- Gérer les marqueurs comme des ensembles
- Organisez la manière dont les marqueurs sont affichés, soit en mode liste, soit regroupés par fichier ou par classe

Accéder

Ruban	Exécuter > Windows > Points d'Arrêt
-------	-------------------------------------

Travailler avec Ensembles de Marqueurs

Les ensembles Marqueur vous permettent de créer des marqueurs sous forme de groupe nommé, que vous pouvez réappliquer à un fichier de code à des fins spécifiques.

Vous pouvez effectuer certaines opérations à partir de la fenêtre Points d'Arrêt & Marqueurs seule, mais pour comprendre et utiliser les marqueurs et les jeux de marqueurs vous devez également afficher le fichier de code approprié dans la 'Visionneuse de code source' (cliquez sur l'élément Classe et appuyez sur F12).

Accéder

Ruban	Exécuter > Windows > Points d'Arrêt : icône de la barre d'outils 
-------	--

Utilisant Ensembles des Marqueur

Action	Détails
Exemple d'utilisation	<p>Vous pouvez créer un ensemble de marqueurs d'enregistrement automatique de méthode pour enregistrer l'action de diverses fonctions dans le code, et un ensemble de marqueurs de capture de pile pour enregistrer la séquence d'appels qui provoquent l'appel de ces fonctions.</p> <p>Vous pouvez ensuite créer diagrammes Séquence à partir des enregistrements sous chaque ensemble.</p>
Créer un Ensemble de Marqueurs	<p>Pour créer un ensemble de marqueurs à partir de la fenêtre Points d'Arrêt & Marqueurs, cliquez sur la flèche déroulante de l'icône  et sélectionnez l'option « Nouvel ensemble ».</p> <p>La boîte dialogue « Nouvel Point d'Arrêt Marqueur s'affiche ; dans le champ « Entrer un nouveau nom d'ensemble », saisissez un nom pour l'ensemble et cliquez sur le bouton Enregistrer.</p> <p>Le nom de l'ensemble s'affiche dans le champ de texte à gauche de l'icône « Définir Options ».</p> <p>Vous pouvez également sélectionner l'option « Enregistrer comme ensemble » dans la liste déroulante « Définir Options » pour créer une copie exacte de l'ensemble actuellement sélectionné, que vous pourrez ensuite modifier.</p>
Accéder Ensembles	<p>Pour accéder à un ensemble de marqueurs, cliquez sur la flèche déroulante dans le champ de texte à gauche de l'icône « Définir Options » et sélectionnez l'ensemble requis dans la liste.</p> <p>Les marqueurs du coffret sont répertoriés dans la fenêtre Points d'Arrêt & Marqueurs .</p> <p>Vous chargeriez normalement un ensemble de marqueurs avant le point auquel une action doit être capturée.</p> <p>Par exemple, pour enregistrer une séquence impliquant un dialogue particulier, lorsque vous commencez le débogage, vous chargez l'ensemble avant d'appeler le dialogue ; une fois que vous avez affiché le dialogue dans l'application, les</p>

	opérations que vous avez marquées sont enregistrées.
Ajouter Marqueurs à Ensemble	<p>Pour ajouter des marqueurs à un ensemble de marqueurs, ajoutez chaque marqueur requis à la ligne de code appropriée dans la « Visionneuse de code source ».</p> <p>Le marqueur est immédiatement ajouté à l'ensemble actuellement répertorié dans la fenêtre Points d'Arrêt & Marqueurs .</p> <p>Chaque marqueur répertorié dans le dialogue possède une case à cocher dans la colonne « Activé » ; les marqueurs nouvellement ajoutés sont automatiquement activés, mais vous pouvez désactiver et réactiver les marqueurs rapidement lorsque vous vérifiez le code.</p>
Stockage des Ensembles	<p>Lorsque vous créez un ensemble de marqueurs, il est immédiatement enregistré dans le modèle ; tout utilisateur utilisant le modèle a accès à cet ensemble.</p> <p>Cependant, l'ensemble par défaut, qui existe toujours pour un modèle, est un espace de travail personnel, n'est pas partagé et est stocké en dehors du modèle.</p>
Supprimer un Marqueur d'un Ensemble	Cliquez-droit sur le marqueur et sélectionnez l'option 'Supprimer Point d'Arrêt '.
Supprimer un ensemble	<p>Si vous n'avez plus besoin d'un ensemble de marqueurs, accédez-y dans la fenêtre Points d'Arrêt & Marqueurs et sélectionnez l'option « Supprimer Sélectionnée » dans la liste déroulante « Définir Options ».</p> <p>Vous pouvez également effacer tous les ensembles de marqueurs définis par l'utilisateur en sélectionnant l'option « Supprimer tous les ensembles » ; une prompt s'affiche pour confirmer la suppression.</p>

Notes

- Ensembles Marqueur sont très simples et flexibles mais, comme ils sont disponibles pour être utilisés par n'importe quel utilisateur du modèle, ils peuvent être facilement corrompus ; tenez compte de ces directives :
 - Lorsque vous nommez un ensemble, utilisez vos initiales dans le nom et essayez d'indiquer son utilisation, afin que d'autres modèles les utilisateurs peuvent reconnaître son propriétaire et son objectif
 - Lorsque vous utilisez un ensemble autre que celui par défaut, évitez les expérimentations excessives afin de ne pas ajouter de nombreux marqueurs ad hoc pour l'ensemble
 - Assurez-vous de savoir quel ensemble de marqueurs est exposé dans la fenêtre Points d'Arrêt & Marqueurs car vous pouvez facilement ajouter par inadvertance des marqueurs à l'ensemble qui ne sont pas pertinents pour le fichier de code.
 - l'ensemble a été créé pour
 - Dans n'importe quel ensemble, si vous avez ajouté des marqueurs qui n'ont pas besoin d'être conservés, supprimez-les pour conserver les objectif de l'ensemble ; ceci est particulièrement vrai pour l'ensemble par défaut, qui peut rapidement s'accumuler marqueurs ad hoc redondants

Contrôler la Session d'Enregistrement

La fenêtre Enregistrer et analyser vous permet de contrôler une session d'enregistrement. Le contrôle dispose d'une barre d'outils et d'une fenêtre d'historique qui affiche l'historique d'enregistrement au fur et à mesure de sa capture. Chaque entrée de cette fenêtre représente une séquence d'appels composée d'un ou plusieurs appels de fonction.

Accéder

Ouvrez la fenêtre Enregistrer et analyser en utilisant l'une des méthodes décrites ici.

Vous devez également ouvrir la fenêtre Analyseur d'Exécution ('Exécuter > Analyseur | Scripts d'Analyseur '), qui liste tous les scripts du modèle ; vous devez sélectionner et activer le script approprié pour l'enregistrement.

Ruban	Exécuter > Outils > Enregistreur > Ouvrir Enregistreur
-------	--

Barre d'Outils d'Enregistreur

Vous pouvez accéder à facilités pour démarrer, arrêter et modérer une session d'enregistrement d'analyse d'exécution via la barre d'outils Enregistrer et analyser.

Accéder

Ruban	Exécuter > Outils > Enregistreur > Ouvrir Enregistreur Explorer > Portails > Afficher la barre d'outils > Enregistrer
-------	--

Boutons

Bouton	Description
	<p>Afficher un menu d'options permettant de définir sur quoi fonctionne la session d'enregistrement.</p> <ul style="list-style-type: none"> Attacher au processus - activée même si aucun script d'analyse n'existe, cette option affiche une dialogue via laquelle vous sélectionnez un processus à enregistrer et une plate-forme de débogage à utiliser ; vous pouvez également sélectionner éventuellement un ensemble de marqueurs d'enregistrement et/ou une Statemachine à utiliser pendant l'enregistrement Générer Diagramme de Séquence à partir de Enregistrement - générer un diagramme Séquence / State à partir de la trace de l' Analyseur d'Exécution Générer Testpoint Diagramme from History - générer un diagramme de domaine Test à partir de la trace de l' Analyseur d'Exécution , qui peut être utilisé avec Testpoint facilité Générer Diagramme de classe à partir de l'historique - générer un diagramme de classe de collaboration à partir de la trace de l' Analyseur d'Exécution , décrivant uniquement les classes et les opérations impliquées dans l'action enregistrée (cas d'utilisation) Générer Graphique d'Appel à partir de l'historique - générer un Graphique d'Appel dynamique à partir de l'historique d'enregistrement, comme vous pouvez le voir dans la disposition d'analyse d'exécution de l'espace de travail du profil VEA ; cela peut être plus utile que le diagramme Séquence pour identifier les piles d'appels uniques impliquées Générer All - générer ensemble les diagrammes Séquence , Testpoint et Collaboration Class à partir de la trace Analyseur d'Exécution Enregistrer comme artefact - crée un élément d'artefact contenant l'historique d'enregistrement actuel, sous le Paquetage actuellement sélectionné dans la fenêtre Navigateur ; si vous faites ensuite glisser cet élément d'artefact sur un diagramme de classe et double-cliquez dessus, l'historique enregistré dans l'artefact est recopié dans la fenêtre Enregistrer et analyser Charger l'historique Séquence à partir d'un fichier - sélectionnez un fichier XML à partir duquel restaurer un historique d'enregistrement précédemment enregistré Enregistrer l'historique Séquence dans un fichier - enregistrez l'historique d'enregistrement dans un fichier XML

	<p>Sélectionnez la profondeur de la pile d'enregistrement pour l'ensemble de marqueurs ; c'est-à-dire le nombre d'images à partir du point où l'enregistrement a commencé.</p>
	<p>Lancez et enregistrez l'application décrite dans le script ; vous pouvez éventuellement sélectionner un ensemble de marqueurs d'enregistrement et/ou une Statemachine à utiliser pendant l'enregistrement.</p> <p>L'icône est activée lorsque le script d'analyse actif est configuré pour le débogage.</p>
	<p>Effectuez un enregistrement manuel ad hoc du thread actuel pendant une session de débogage.</p> <p>Utilisez cette fonction avec les boutons « étape » du débogueur ; chaque fonction appelée en raison d'une commande d'étape est enregistrée dans la fenêtre d'historique.</p> <p>L'icône est activée si aucun enregistrement n'est en cours et que vous êtes actuellement à un point d'arrêt (c'est-à-dire en cours de débogage).</p>
	<p>Effectuez un enregistrement automatique ad hoc pendant une session de débogage.</p> <p>Lorsque vous cliquez sur cette icône, l'analyseur commence l'enregistrement et ne s'arrête pas tant que le programme n'est pas terminé, que vous n'arrêtez pas le débogueur ou que vous n'avez pas cliqué sur l'icône Arrêter.</p> <p>Cette icône est activée si aucun enregistrement n'est en cours et que vous êtes actuellement à un point d'arrêt (c'est-à-dire en cours de débogage).</p>
	<p>Entrez dans une fonction, enregistrez l'appel de fonction dans la fenêtre Historique et revenez en arrière.</p> <p>Activé pour l'enregistrement manuel uniquement.</p>
	<p>Arrêtez l'enregistrement de la trace d'exécution.</p>
	<p>Affichez la dialogue « Synchroniser Modèle » à travers laquelle vous pouvez synchroniser le modèle avec les fichiers de code générés lors d'une opération de profil d'enregistrement.</p>

Travailler avec Historique d'Enregistrement

Vous pouvez effectuer un certain nombre d'opérations sur ou à partir des résultats d'une session d'enregistrement, en utilisant le menu contextuel de la fenêtre Enregistrer et analyser.

Options

Option	Action
Afficher la source de l'appelant	Affichez le code source, dans la Visionneuse de code source , pour la méthode appelant la séquence.
Afficher la source pour Callee	Affichez le code source, dans la Visionneuse de code source , pour la méthode appelée par la séquence.
Générer Diagramme pour Séquence sélectionnée	Générer un diagramme Séquence pour une séquence unique sélectionnée dans l'historique d'enregistrement.
Générer Diagramme de Séquence	Générer un diagramme Séquence incluant toutes les séquences de l'historique d'enregistrement.
Effacer	Effacer l'historique d'enregistrement actuellement affiché dans la fenêtre Enregistrer et analyser.
Enregistrer Historique d'Enregistrement dans un fichier	Enregistrez l'historique d'enregistrement dans un fichier XML. Une fenêtre de navigateur s'affiche, dans laquelle vous spécifiez le chemin d'accès et le nom du fichier XML.
Charger Historique d'Enregistrement à partir du fichier	Chargez un historique d'enregistrement précédemment enregistré à partir d'un fichier XML. Une fenêtre de navigateur s'affiche, dans laquelle vous spécifiez le chemin d'accès et le nom du fichier XML à charger.
Désactiver tous les appels	Désactivez tous les appels répertoriés dans la fenêtre Enregistrer et analyser.
Désactiver cet appel	Désactiver l'appel sélectionné.
Désactiver cette méthode	Désactiver la méthode sélectionnée.
Désactiver cette classe	Désactiver la classe sélectionnée.
Désactiver tous les appels en dehors de cet appel	Désactivez tous les appels répertoriés dans la fenêtre Enregistrer et analyser, à l'exception de l'appel sélectionné.
Activer tous les appels	Activez chaque appel répertorié dans la fenêtre Enregistrer et analyser.
Activer cet appel	Activer l'appel sélectionné.
Activer cette méthode	Activer la méthode sélectionnée.

Activer cette classe	Activer la classe sélectionnée.
Aide	Afficher la rubrique d'aide de la fenêtre Enregistrer et analyser.

Démarrer Enregistrement

Lorsque vous enregistrez un flux d'exécution sous forme de diagramme Séquence , démarrez l'enregistrement en sélectionnant l'icône « Enregistrement » dans la barre d'outils de la fenêtre Enregistrer et analyser. La dialogue « Enregistrer » s'affiche avec les options d'enregistrement définies par défaut, c'est-à-dire le Point d'Arrêt et le jeu Marqueurs actuels, les filtres définis dans le script d'analyse actuel et le mode d'enregistrement de base.

Accéder

Ruban	Exécuter > Outils > Enregistreur > Ouvrir Enregistreur : 
-------	--

Options d'enregistrement Dialogue

Champ/Bouton	Détail
Ensemble Enregistrement	Les marqueurs Enregistrement déterminent ce qui est enregistré. Si vous disposez d'un ensemble d'enregistrements à utiliser, cliquez sur la flèche déroulante et sélectionnez-le.
Filtres supplémentaires	Les filtres sont utilisés par le débogueur pour exclure les appels de fonctions correspondants de l'historique d'enregistrement. Les filtres Enregistrement sont définis dans le script Analyzer. Dans le champ « Filtres supplémentaires », vous pouvez ajouter d'autres filtres pour cet exécuter spécifique. Si vous spécifiez plusieurs filtres, séparez-les par un point-virgule.
Mode Enregistrement de base	En mode de base, le débogueur enregistre un historique des appels de fonction effectués par le programme chaque fois qu'il rencontre un marqueur d'enregistrement approprié.
Suivre les instances des classes nommées	En mode Suivi des instances, le débogueur capture également la création des instances des classes que vous spécifiez. Il inclut ensuite ces informations dans l'historique. Le diagramme Séquence résultant peut alors afficher les lignes de vie pour chaque instance de cette classe avec, le cas échéant, les appels de fonction liés à la ligne de vie.
Suivre les transitions State	L'enregistrement peut également capturer les changements d' State à l'aide d'un diagramme Statemachine spécifié. Le diagramme Statemachine doit exister en tant qu'enfant d'une classe. L' Analyseur d'Exécution capture les instances de cette classe et calcule l' State de chaque instance chaque fois qu'une fonction de la séquence d'enregistrement en cours revient.
OK	Cliquez sur ce bouton pour démarrer le débogueur.

Parcourez Appels de Fonction

La commande « Pas à pas » peut être exécutée en cliquant sur le bouton Pas à pas dans la barre d'outils de la fenêtre Enregistrer et analyser.

Vous pouvez également appuyer sur Maj+F6 ou sélectionner l'option de ruban « Exécuter > Exécuter > Entrer ».

La commande « Step Through » provoque l'exécution d'une commande « Step Into » ; si une fonction est détectée, cet appel de fonction est enregistré dans la fenêtre Historique.

Le Débogueur sort alors et le processus peut être répété.

Ce bouton vous permet d'enregistrer un appel sans avoir à accéder à une fonction ; le bouton n'est activé que lorsqu'il se trouve à un point d'arrêt et en mode d'enregistrement manuel.

Marqueurs d'Enregistrement Nested

Lorsqu'un marqueur d'enregistrement est rencontré pour la première fois, l'enregistrement démarre à l'image de pile actuelle et continue jusqu'à ce que l'image apparaisse, enregistrant des images supplémentaires jusqu'à la profondeur définie dans la barre d'outils Enregistrement . Considérez cette séquence d'appel :

```
A -> B -> C -> D -> E -> F -> G -> H -> I -> J -> K -> L -> M -> N -> O -> P -> Q -> R -> S -> T -> U -> V -> W -> X -> Y -> Z
```

Si vous définissez un marqueur d'enregistrement sur K et définissez la profondeur d'enregistrement sur 3, cela enregistrera la séquence d'appel :

```
K -> L -> M
```

Si vous souhaitez également enregistrer les appels X, Y et Z dans le cadre du diagramme Séquence , vous placeriez un autre marqueur d'enregistrement à X et l'analyseur enregistrerait :

```
K -> L -> M -> X -> Y -> Z
```

Cependant, lorsque l'enregistrement se termine pour le composant XYZ (l'image X est supprimée), l'enregistrement reprend lorsque l'image M de la séquence KLM est réintégrée. L'utilisation de cette technique peut aider là où les informations du diagramme enregistré seraient exclues en raison de la profondeur de la pile, et elle vous permet de vous concentrer sur les zones particulières à capturer.

Génération de Diagrammes Séquence

Cette rubrique décrit ce que vous pouvez faire avec l'enregistrement d'une session d'analyse d'exécution.

Accéder

Ruban	Exécuter > Outils > Enregistreur > Ouvrir Enregistreur
-------	--

Référence

Action	Détail
Générer un diagramme	<p>Sélectionnez le Paquetage approprié dans la fenêtre Navigateur , dans lequel stocker le diagramme Séquence .</p> <p>Pour créer le diagramme à partir de toutes les séquences enregistrées, vous pouvez soit :</p> <ul style="list-style-type: none"> • Cliquez sur l'icône « Menu Enregistreur » () dans la barre d'outils de la fenêtre Enregistrer et analyser, et sélectionnez l'option « Générer Diagramme de Séquence à partir de Enregistrement », ou • Cliquez-droit sur le corps de la fenêtre et sélectionnez l'option ' Générer Diagramme de Séquence ' <p>Pour créer le diagramme à partir d'une seule séquence, vous pouvez soit :</p> <ul style="list-style-type: none"> • Cliquez sur l'icône « Menu Enregistreur » () dans la barre d'outils de la fenêtre Enregistrer et analyser, et sélectionnez l'option « Générer Diagramme de Séquence à partir de Enregistrement », ou • Cliquez-droit sur la séquence et sélectionnez l'option ' Générer Diagramme à partir de Séquence Sélectionnée '
Enregistrer une séquence enregistrée dans un fichier XML	<p>Cliquez sur la séquence, cliquez sur l'icône « Menu Enregistreur » () dans la barre d'outils de la fenêtre Enregistrer et analyser, et sélectionnez l'option « Enregistrer l'historique Séquence dans un fichier ».</p>
Accéder à un fichier XML de séquence existant	<p>Soit:</p> <ul style="list-style-type: none"> • Cliquez sur le  dans la barre d'outils de la fenêtre Enregistrer et analyser, puis sélectionnez l'option « Charger l'historique Séquence à partir du fichier », ou • Cliquez-droit sur une zone vide de l'écran et cliquez sur l'option « Charger Séquence à partir du fichier » <p>La dialogue « Ouvrir Windows » s'affiche, à partir de laquelle vous sélectionnez le fichier à ouvrir.</p>

Utiliser pour

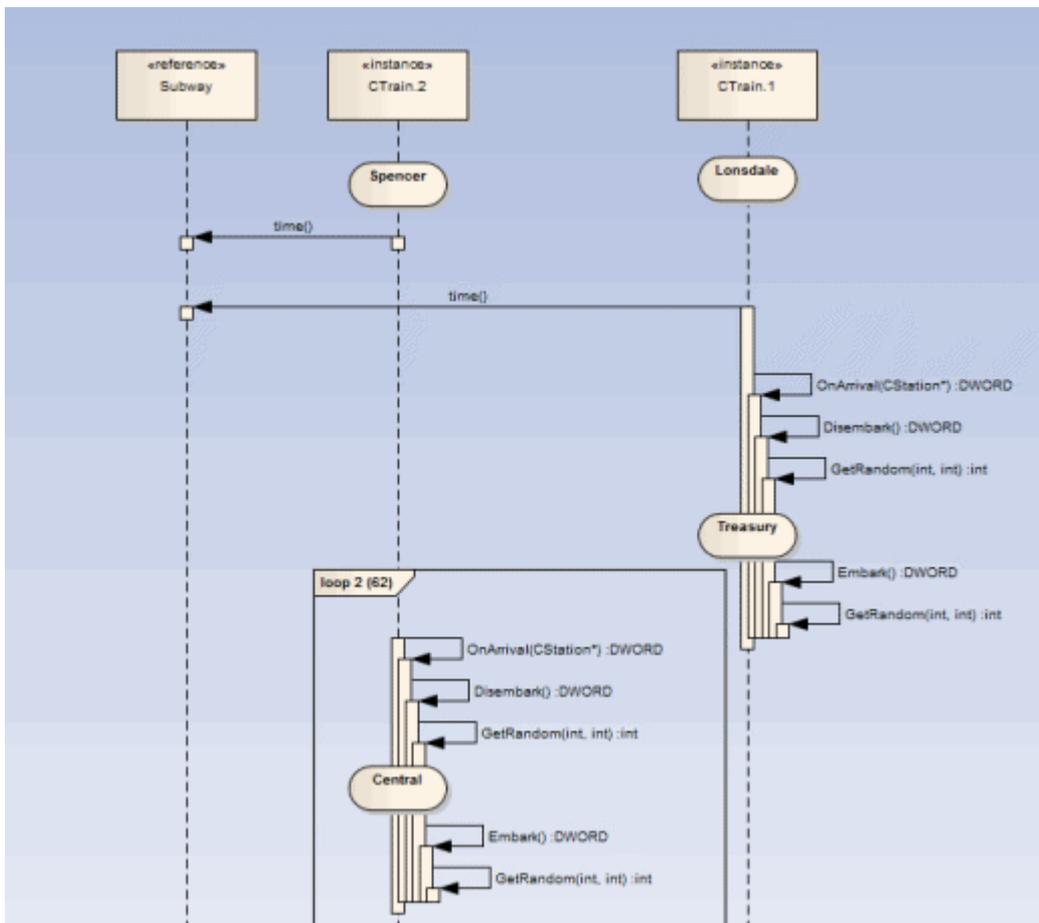
- Générer un diagramme Séquence à partir d'une session d'analyse d'exécution enregistrée, pour :
- toutes les séquences enregistrées ou
- une seule séquence dans la séance
- Sauvegarder la séquence enregistrée dans un fichier
- Récupérez l'enregistrement sauvegardé et chargez-le dans la fenêtre Enregistrer et analyser

Rapportage les transitions State

Cette section décrit comment générer diagrammes Séquence qui montrent les transitions d'état lors de l'exécution d'un programme.

Utiliser pour

Générer diagrammes Séquence qui signalent les transitions définies par l'utilisateur dans l'état pendant l'exécution d'un programme (comme indiqué dans l'exemple diagramme généré)



Sujet

Créez une Statemachine sous la classe à signaler.

Définissez les contraintes par rapport à chaque State pour définir le changement d'état à signaler.

Rapportage d'une Statemachine

L'Analyseur d'Exécution peut enregistrer un diagramme Séquence, nous le savons. Ce que vous ne savez peut-être pas, c'est qu'il peut utiliser en même temps une Statemachine pour détecter les transitions State qui pourraient survenir en cours de route. Ces States sont représentés à un instant donné sur la ligne de vie de l'objet. Les transitions sont également apparentes à partir des lignes de vie. Toute transition invalide ou illégale sera mise en évidence par une bordure rouge. Jetez-y un œil.

Processus

Tout d'abord, vous modélisez une Statemachine pour l'élément de classe approprié.

Vous composez ensuite les expressions qui définissent chaque State en utilisant l'onglet « Contraintes » de chaque State.

Ces expressions simples sont formées à l'aide de noms d'attributs issus du modèle de classe et de la base de code réelle. Ce ne sont pas des instructions OCL. Chaque expression doit apparaître sur une ligne distincte.

```
m_strColor == "Bleu"
```

Vous utilisez ensuite la fenêtre Enregistreur pour lancer le débogueur.

Le bouton Exécuter de la fenêtre Enregistreur est différent du bouton des autres barres d'outils du débogueur.

La fenêtre Enregistreur vous permettra de rechercher une Statemachine si vous ne connaissez pas Statemachine nom. La dialogue « State Transition » présente une liste de Statemachines pour l'ensemble du modèle, dans laquelle vous pouvez localiser et sélectionner le diagramme approprié (voir l'exemple).

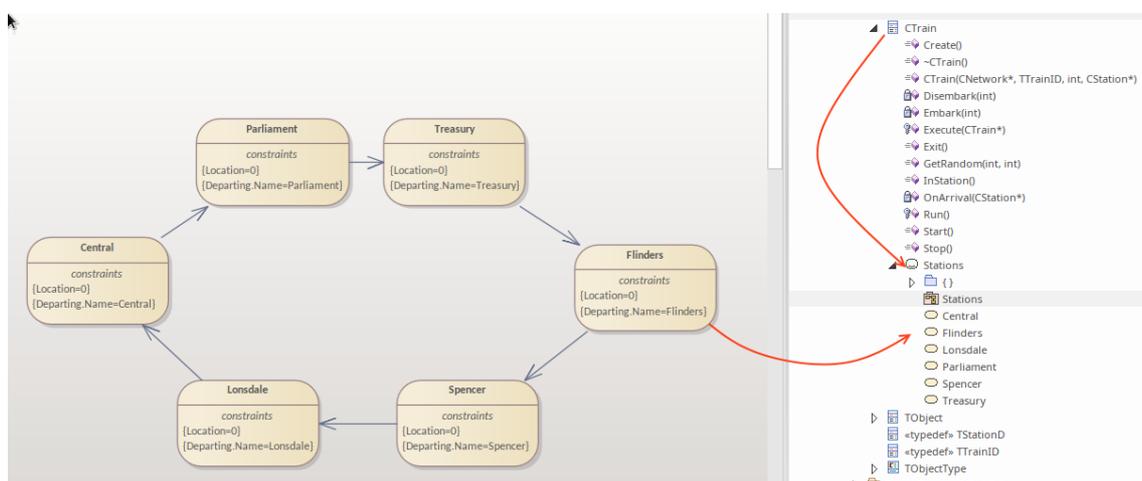
Lorsque vous générez le diagramme Séquence, il décrit non seulement la séquence mais également les changements d'Etat aux différents points de la séquence ; chaque instance de classe participant au processus de détection est affichée avec sa propre ligne de vie.

Exemple

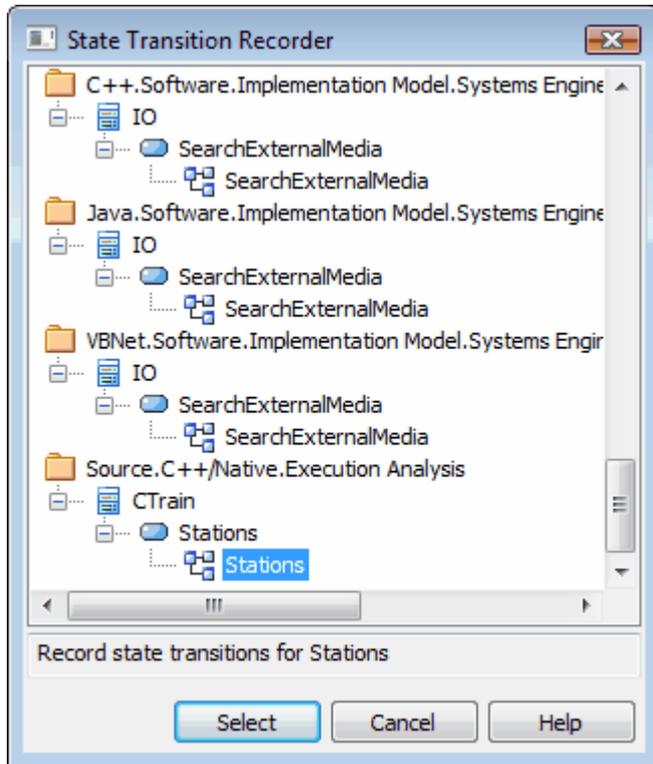
La Statemachine des stations montre les différents States du système de métro Melbourne Underground Loop.

Un train circulant sur le réseau de métro peut être arrêté à n'importe laquelle des stations représentées sur la Statemachine.

La Statemachine des stations est un enfant de la classe CTrain.



Lorsque vous recherchez le diagramme dans la dialogue « Enregistreur de transition State », la hiérarchie affiche uniquement le Paquetage racine, la classe parent et la sous-machine enfant et diagramme ; aucun autre composant du modèle n'est répertorié.



Enregistrement et Mappage Changements de State

Cette rubrique explique comment définir des contraintes pour chaque State dans la Statemachine sous une classe, pour définir le changement d'état à enregistrer.

Exemple

Cet exemple de dialogue « Propriétés » State concerne l' State appelé Parlement ; l'onglet « Contraintes » est ouvert pour montrer comment l' State est lié à la classe CXTrain.

Un State peut être défini par une seule contrainte ou par plusieurs ; dans l'exemple, l' State Parlementaire a deux contraintes :

Constraint	Type	Status
Location=0	Invariant	Approved
Departing.Name=Parliament	Invariant	Approved

Les valeurs des contraintes ne peuvent être comparées que pour les types élémentaires, enum et string

La classe CXTrain possède un membre appelé Location de type int et un membre appelé Departing.Name de type CString ; cette contrainte signifie que cet State est évalué à True lorsque :

- une instance de la classe CXTrain existe et
- sa variable membre Location a la valeur 0 et
- la variable membre Departing.Name a la valeur Parliament

Opérateurs dans les contraintes

Il existe deux types d'opérateurs que vous pouvez utiliser sur les contraintes pour définir un State :

- Les opérateurs logiques AND et OR peuvent être utilisés pour combiner des contraintes
- Les opérateurs d'équivalence {= et !=} peuvent être utilisés pour définir les conditions d'une contrainte

Toutes les contraintes pour un State sont soumises à une opération AND sauf indication contraire ; vous pouvez utiliser l'opération OR à la place, vous pouvez donc réécrire les contraintes dans l'exemple comme suit :

Emplacement=0 OR

Emplacement=1 AND

Départ.Nom!=Central

Voici quelques exemples d'utilisation des opérateurs d'équivalence :

Départ.Nom!=Central AND

Emplacement!=1

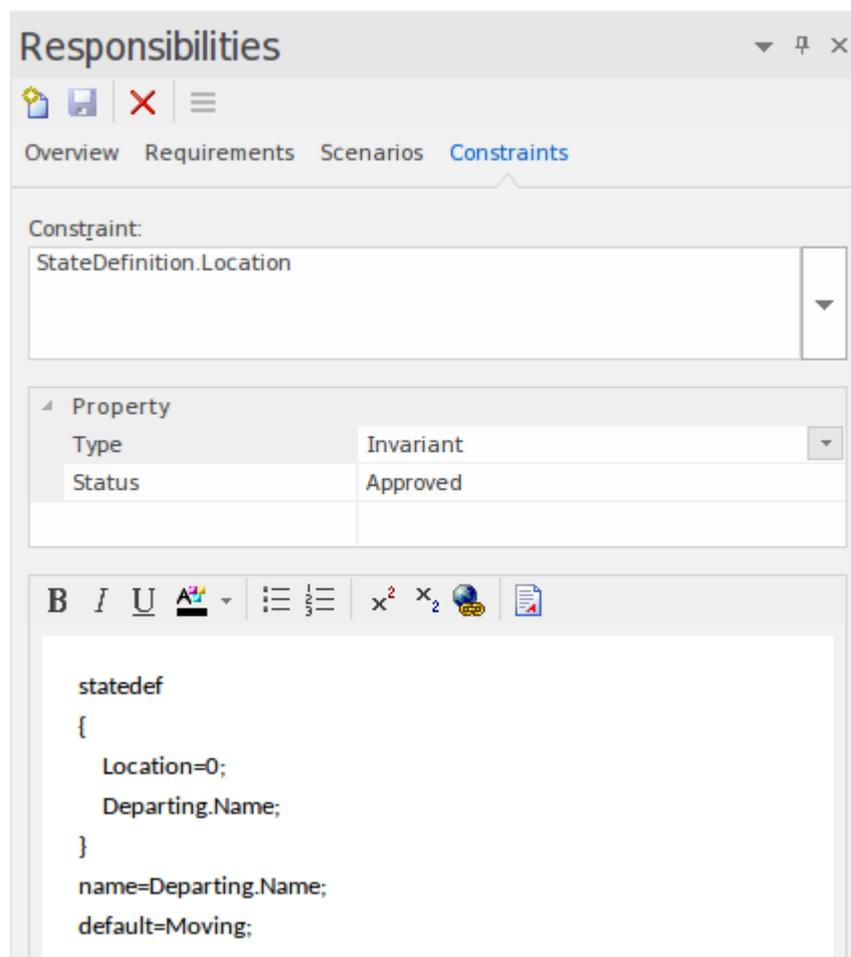
Notes

- Les guillemets autour des chaînes sont facultatifs ; la comparaison des chaînes est toujours sensible à la casse pour déterminer la véracité d'une contrainte

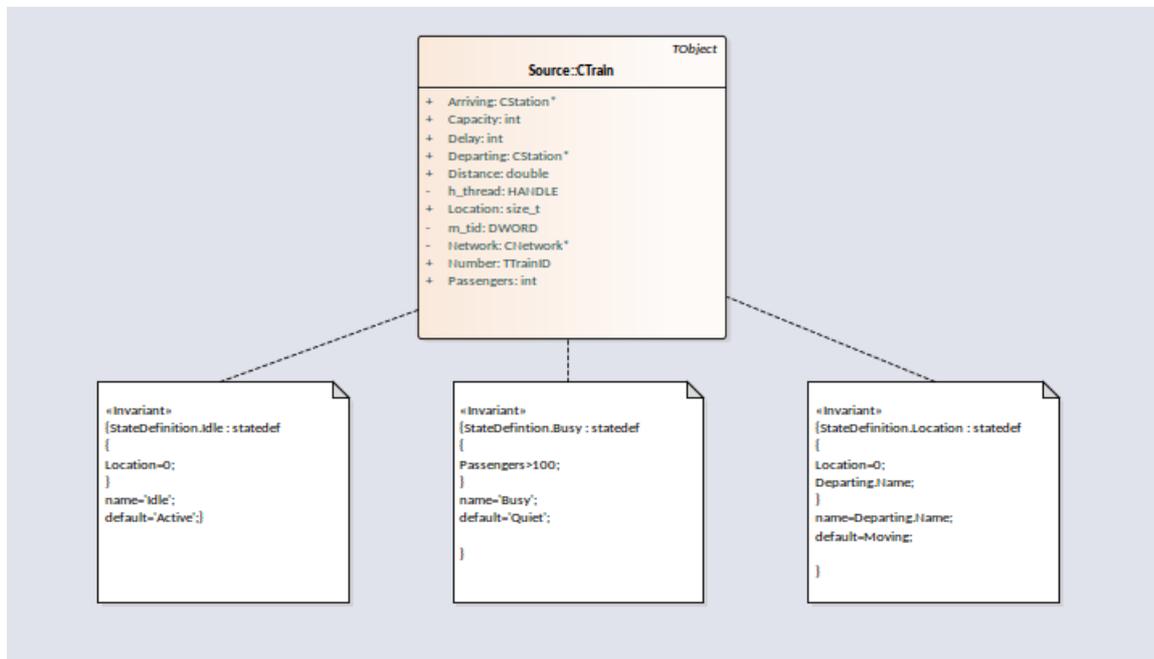
Analyseur de State

L'Analyseur de State est une fonctionnalité qui permet d'analyser, de détecter et d'enregistrer les états des instances d'une Classe. La fonctionnalité fonctionne en combinant une définition d'état (définie sur une Classe comme une contrainte) et des marqueurs appelés points State . Elle est disponible pour tous les langages pris en charge par l'Analyseur d'Exécution , y compris Microsoft.NET, Mono, Java et le C++ natif.

Nous commençons par sélectionner une classe et composer notre définition d'état.



Nous pouvons obtenir une image de toutes les définitions d'état que nous avons définies en plaçant la classe sur un diagramme et en établissant un lien vers les notes de classe qui sont elles-mêmes liées à une contrainte de définition d'état particulière. Nous expliquons comment procéder dans une section ultérieure.



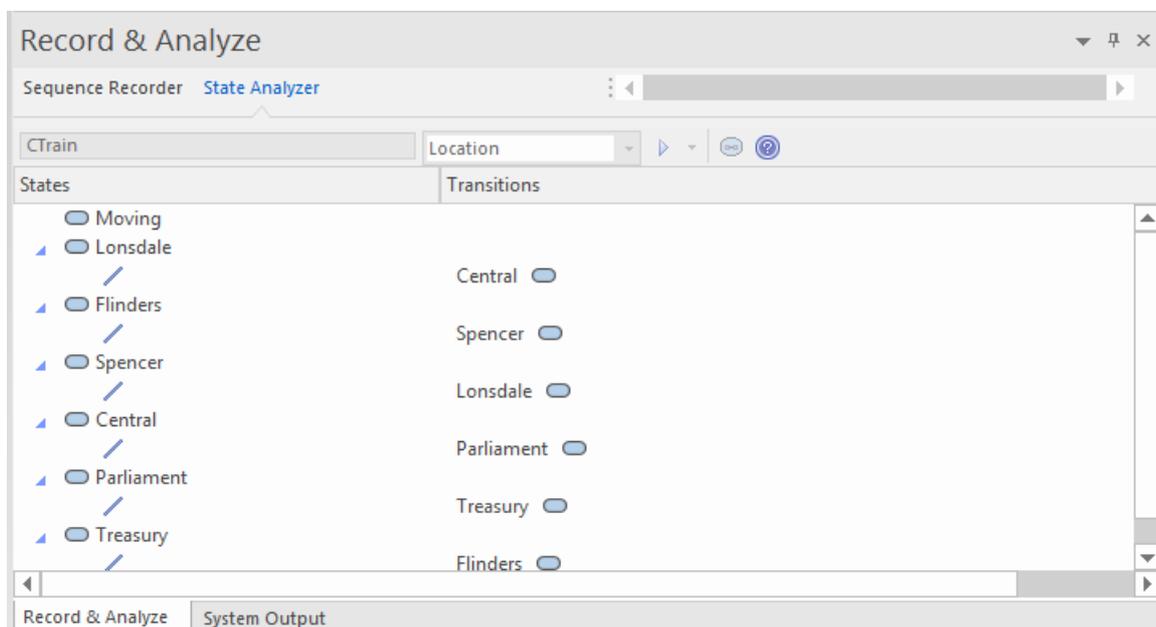
Les points State sont définis en plaçant un ou plusieurs marqueurs dans le code source pertinent.

```

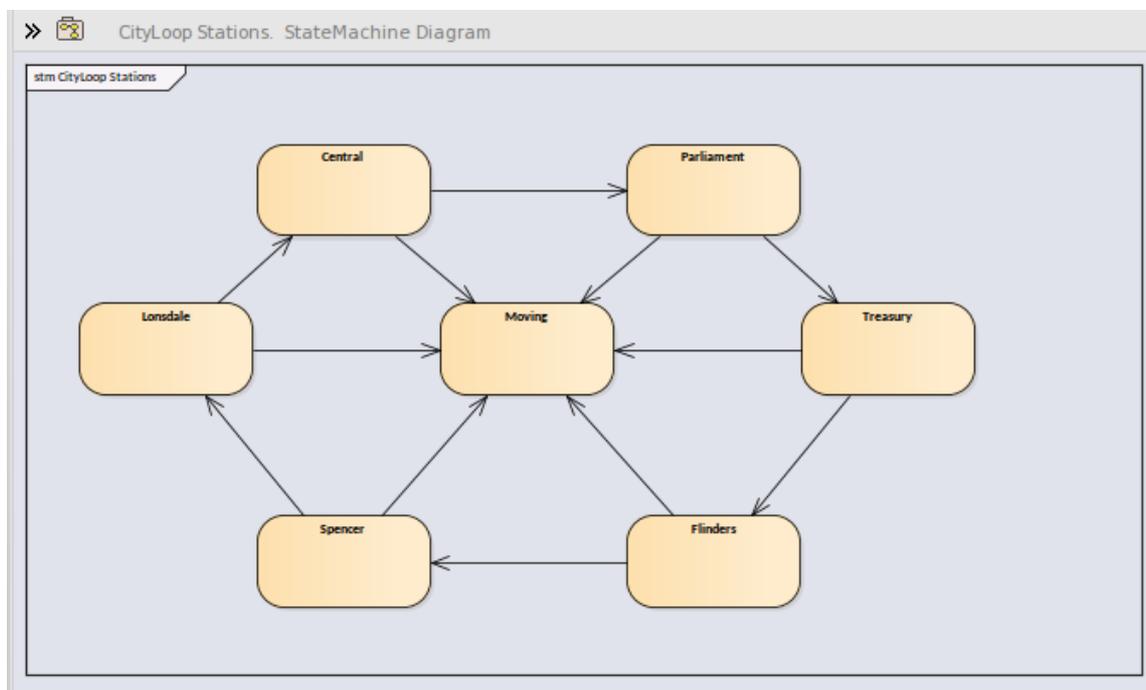
73 DWORD  CTrain::OnArrival( CStation* S)
74 {
75     Departing = S;
76     Location = 0;
77     Delay = (Disembark(GetRandom()) + Embark(GetRandom()));
78     DWORD ScheduleTime = Network->TimeAtStation(Departing);
79     if(Delay > (int)ScheduleTime)
80         return Delay;
81     return ScheduleTime;
82 }

```

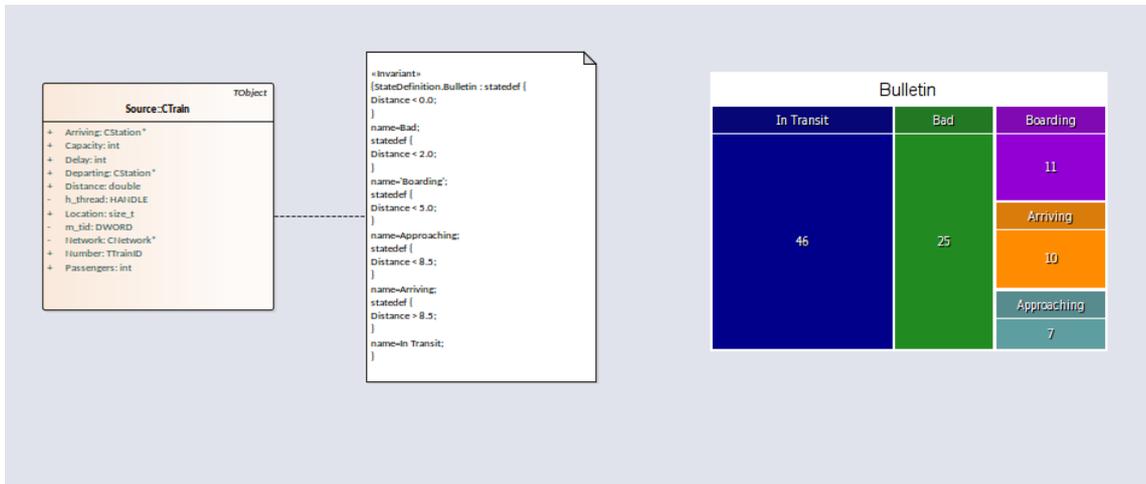
Le programme à analyser est exécuté à l'aide du contrôle Analyseur de State . Lorsque l' Analyseur d'Exécution rencontre un point State , l'instance courante de la classe est analysée. Lorsque le domaine valeur de l'instance correspond à la définition de l'état, un état est enregistré. Chaque fois que l'instance varie, de nouveaux états sont ainsi détectés. Le contrôle liste chaque état au fur et à mesure qu'il est découvert. Sous chaque état, le contrôle liste l'ensemble discret des transitions vers d'autres états effectuées par les instances de la classe.



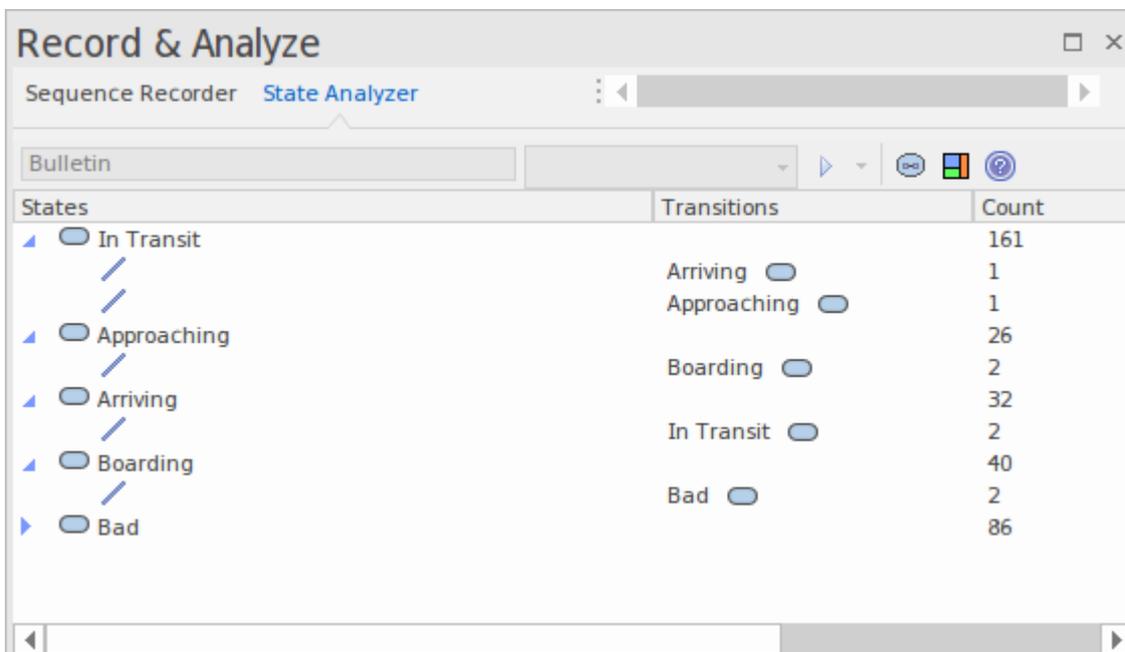
Les informations peuvent être utilisées pour créer une Statemachine .



En utilisant les mêmes informations, nous pouvons facilement produire une Carte de Chaleur . Cet exemple montre une classe « Train », sa définition State « Bulletin » (sous forme de note liée) et la Carte de Chaleur qu'elle a produite. Les chiffres de la carte sont des pourcentages. À partir de la carte, nous pouvons observer que les trains étaient dans l'état « En transit » 46 % du temps.



Il s'agit de l'analyse de la définition State du « Bulletin » qui a produit notre Carte de Chaleur .



Accéder

Ruban	Exécuter > Outils > Enregistreur > Ouvrir Enregistreur > Analyseur de State Conception > Élément > Éditeurs > Contraintes
-------	--

Définitions de State

Définitions de State sont composées dans les propriétés de contrainte d'un élément de classe. Le type de contrainte doit être nommé *StateDefinition.name* , où 'name' est le titre de votre choix pour la définition. Ces titres sont répertoriés dans la zone de liste déroulante de l' Analyseur de State chaque fois qu'une classe est sélectionnée. Vous sélectionnez une seule définition dans cette zone de liste déroulante avant d'exécuter le programme. La définition State de notre exemple est nommée 'StateDefinition.Location'. Elle définit les états en fonction de l'emplacement des instances de la classe CTrain.

Définitions de State sont composées d'une ou plusieurs spécifications. Chaque spécification d'état commence par le mot-clé « statedef » qui est ensuite suivi d'une ou plusieurs instructions. Les instructions définissent les contraintes qui décrivent l'état, et éventuellement une variable dont valeur peut être utilisée pour nommer l'état. Les instructions sont placées entre accolades et se terminent par un point-virgule comme indiqué :

```
statedef {  
Location=0;  
Départ.Nom;  
}
```

Nommer les états à l'aide de variables

Dans cet exemple, 'Location' est une constante et 'Departing.name' est une variable. Une instruction supplémentaire suit les contraintes et indique le nom de l' State à attribuer à partir de la variable valeur . Voici la définition avec la directive de nommage.

```
statedef {  
Location=0;  
Départ.Nom;  
}  
name=Departing.Name;
```

Nommer les états à l'aide de littéraux

Dans cet exemple, la définition State contient uniquement des constantes et l'état est nommé à l'aide d'un littéral.

```
statedef {  
Location=100;  
}  
nom='Central';
```

Une définition State unique définissant plusieurs spécifications State .

```
statedef {  
Passagers > 100 ;  
}  
nom=Occupé;  
statedef {  
Passagers >= 50 ;  
}  
nom=Silencieux;  
statedef {  
Passagers < 50 ;  
}  
nom=Très calme;  
statedef {  
Passagers = 0;
```

```
}
nom=Inactif;
```

State par défaut

Une définition State peut spécifier un état par défaut « fourre-tout » qui décrira l'état d'une instance lorsqu'aucun autre état n'est vrai. Vous définissez un état par défaut pour la définition avec une instruction ressemblant à ceci :

```
statedef {
Location=0;
Départ.Nom;
}
name=Departing.Name;
par défaut = En mouvement ;
```

Dans cet exemple, pendant que l'exécution est en cours, toute instance détectée ayant un attribut « Emplacement » différent de zéro sera enregistrée comme étant dans l'état « En mouvement ».

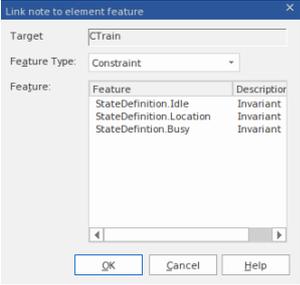
Vous pouvez choisir d'exclure l'enregistrement de l'état par défaut en désactivant l'option « Inclure l'état par défaut » dans le menu déroulant de la barre d'outils Analyseur de State . Cela exclurait l'enregistrement des transitions vers tout état « par défaut ».

Création Notes sur un élément de classe qui affichent Définitions de State

Cette section décrit comment créer le diagramme de classe qui montre toutes les Définitions de State définies pour la classe.

Actes

Afficher un diagramme de classe	Ouvrez un diagramme de classe existant ou créez-en un nouveau.
Créer un lien vers l'élément Class	Faites glisser la classe qui vous intéresse sur le diagramme en tant que lien.
Créer un élément note	Créez un élément note sur le diagramme et liez-le à la classe.
Associez la note à la définition State	Sélectionnez le lien entre la Note et la classe et, en utilisant son menu contextuel, sélectionnez l'option « Lier Note à Fonctionnalité de l'élément ».
Choisissez la définition à	Dans la dialogue des éléments, choisissez « Contraintes » dans la liste déroulante.

<p>afficher sur la Note</p> 	<p>Toutes Définitions de State définies seront répertoriées pour que vous puissiez faire votre choix.</p>
<p>Répéter</p>	<p>Répétez la procédure pour toutes les autres Définitions de State de la classe.</p>

Synchronisation

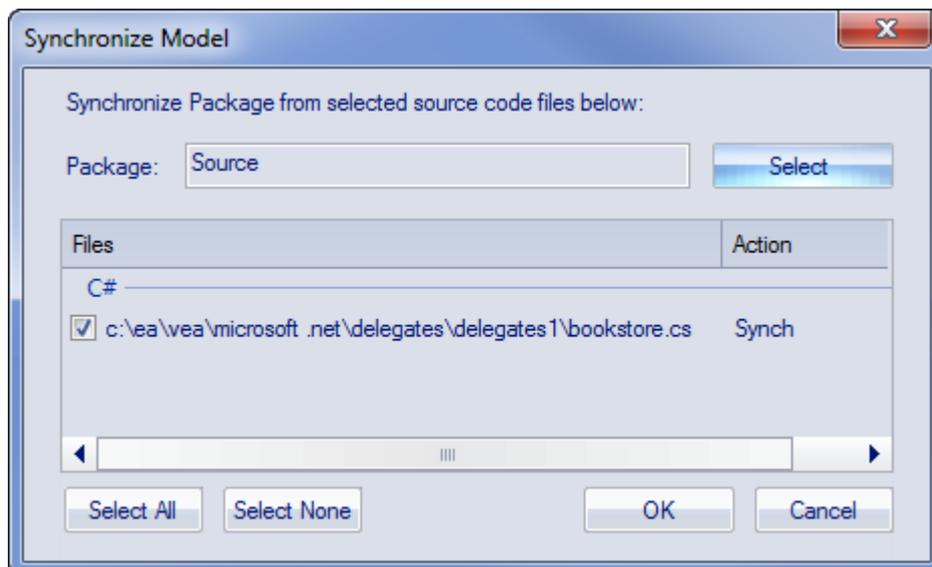
L'enregistrement produit un certain nombre d'éléments, l'historique d'enregistrement étant le principal. Enregistrement identifie également un ensemble de fichiers de code source. Cet ensemble peut être utilisé pour produire diagrammes de classes et de domaines Test, mais peut également être utilisé pour synchroniser votre modèle.

Un modèle synchronisé fournit une navigation rapide et précise entre les éléments diagramme et le modèle de classe.

Accéder

Ruban	Exécuter > Outils > Enregistreur > Ouvrir Enregistreur > Bouton  de la barre d'outils
Menu Contexte	Cliquez-droit sur la fenêtre Enregistrer & Analyser Synchroniser Modèle avec Code Source

Synchroniser Modèle



Champ/Bouton	Action
Paquetage	Cliquez sur le bouton Sélectionner et sélectionnez le Paquetage cible dans lequel effectuer la rétro-ingénierie des fichiers de code.
Fichiers/ Action	Répertorie les fichiers identifiés lors d'un ou plusieurs enregistrements. L'action appropriée est indiquée à côté de chaque fichier.
Sélectionner tout	Cliquez sur ce bouton pour sélectionner la case à cocher correspondant à chaque fichier dans la liste « Fichiers ».
Ne rien sélectionner	Cliquez sur ce bouton pour décocher la case correspondant à chaque fichier dans la

	liste « Fichiers ».
OK	Cliquez sur ce bouton pour démarrer l'opération. La progression de la synchronisation s'affiche.
Annuler	Cliquez sur ce bouton pour interrompre la synchronisation et fermer le dialogue .

Visualiser State Exécuter

Vous pouvez enregistrer les transitions d'état d'un objet unique en prenant plusieurs Instantanés de l'état exécuter de l'objet à des moments clés de sa durée de vie. Pour ce faire, faites simplement glisser la variable locale ou membre sur un diagramme Object .

Object Établi

L' Object Établi est un outil de débogage Enterprise Architect qui vous aide à créer des objets à partir de votre modèle de classe. L' Établi permet à plusieurs instances de n'importe quelle classe de coexister dans la même session. Chaque Object peut servir de cible à une méthode que vous souhaitez invoquer. Ils peuvent également participer en tant que paramètres aux méthodes que vous invoquez. L' Object Établi est pris en charge pour les plates-formes Java et Microsoft .NET .

Tâches d' Établi

Tâche
Fournit un guide et les exigences pour l'utilisation de l' Object Établi .
Explique ce que sont les objets Établi et comment les créer.
Explique comment exécuter des méthodes sur un Object Établi et fournit des informations sur la transmission d'arguments.
Explique le déroulement de l'exécution d'une méthode à l'aide du Débogueur .
Explique comment enregistrer une méthode et produire un diagramme Séquence .
Explique comment supprimer un Object Établi une fois que vous avez terminé de l'utiliser.
Explique comment arrêter le Débogueur et fermer l' Établi une fois que vous avez terminé de l'utiliser.

Utilisant l'Établi

L'utilisation de l'Établi Object est simple. À partir de votre modèle de classe, sélectionnez les classes à utiliser dans l'atelier et faites-les glisser individuellement dans la fenêtre Établi . Vous devrez peut-être choisir un constructeur s'il en existe plusieurs, puis donner simplement un nom à la variable. L'atelier Object prépare le runtime requis, charge les modules requis et instancie les objets pour vous. L'exécution d'une méthode consiste à sélectionner dans une liste. Des paramètres peuvent être saisis si nécessaire. Les objets Établi eux-mêmes peuvent être utilisés comme paramètres soit seuls, soit sous forme de tableaux object .

Accéder

Ruban	Exécuter > Outils > Tester > Ouvrir Object Établi
-------	---

Exigences du script d'analyse

Un script d'analyse configuré pour le débogage est requis. Il doit spécifier les informations suivantes :

- Le débogueur adapté à votre projet
- Pour Microsoft .NET , l'emplacement de l'assembly qui sera hébergé par l' Object Établi
- Pour Java, l'emplacement du JDK et les chemins de classe supplémentaires à utiliser

Liste de contrôle

- Sélectionnez la classe Établi souhaitée et appuyez sur F12 ; le code source doit être affiché dans un éditeur de code
- Appuyez sur Maj+F12 pour générer le projet ; la sortie de la génération devrait indiquer une compilation réussie

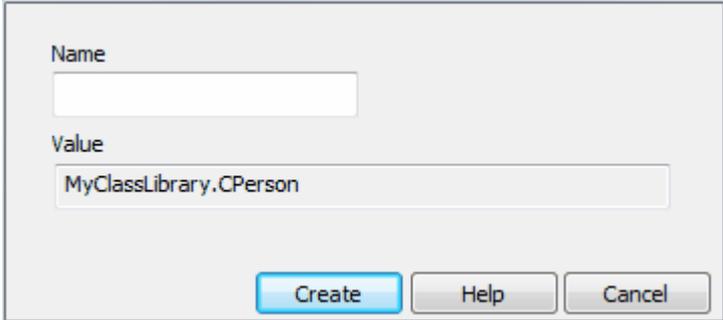
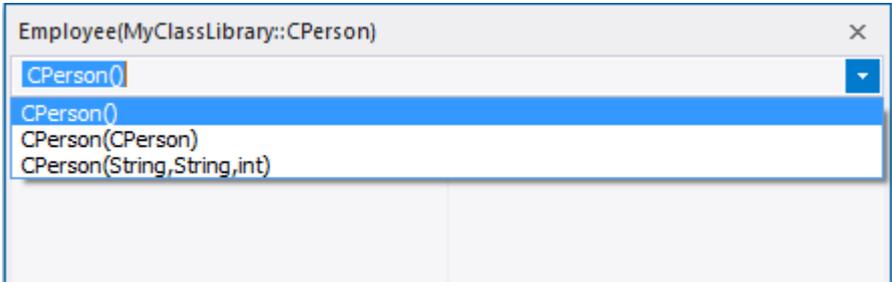
Créer des objets

Cette rubrique explique comment créer une instance d'atelier pour une classe à partir de votre modèle.

Accéder

Ruban	Exécuter > Outils > Tester > Ouvrir Object Établi
Raccourcis clavier	Ctrl+Maj+J
Autre	Faites glisser une classe directement de la fenêtre Navigateur vers la fenêtre Établi

Tâches

Tâche	Détail
Créer un Object sur l'Établi	<p>Sélectionnez la Classe dans la fenêtre Navigateur et faites-la glisser sur la fenêtre Établi .</p> <p>La dialogue ' Établi ' s'affiche.</p>  <p>Type un nom pour la nouvelle instance. Le nom doit être unique pour l' Établi .</p> <p>Cliquez sur le bouton Créer.</p>
Choisir un constructeur	<p>La dialogue « Constructeur » s'affiche et propose un choix de constructeur.</p>  <p>Sélectionnez le constructeur dans la liste déroulante.</p>
Entrer les paramètres	<p>Fournir des valeurs pour les paramètres du constructeur sélectionné :</p> <ul style="list-style-type: none"> Chaînes comme arguments - Entourez les valeurs de guillemets lorsque cela est

	<p>approprié ou lorsque la valeur entrerait en conflit avec le nom d'un objet Établi</p> <ul style="list-style-type: none">• Objets comme arguments - Entrez le nom de l' objet Établi• Les arguments du tableau String prennent des valeurs de texte séparées par des virgules : <p>un, deux, trois, « un livre », « un plus grand livre »</p> <ul style="list-style-type: none">• Les tableaux Object comme arguments prennent des noms object séparés par des virgules ; fournissez les objets Établi nommés séparés par des virgules, par exemple : <p>Tom, Dick, Harry</p>
Invoquer le constructeur	Cliquez sur le bouton Invoke pour créer l'instance. L' object est reconnaissable à son nom dans la fenêtre Établi .

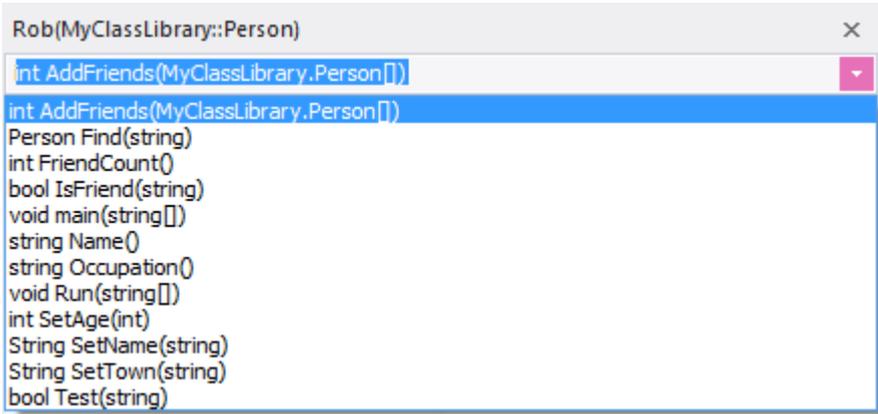
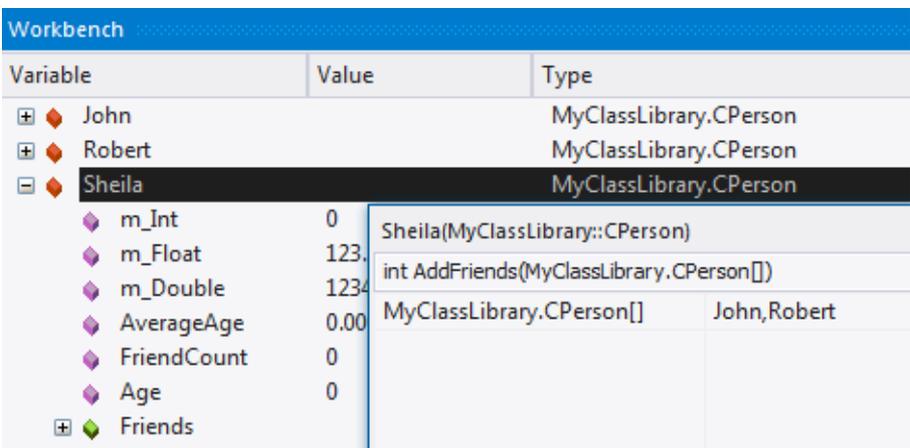
Invocation Méthodes

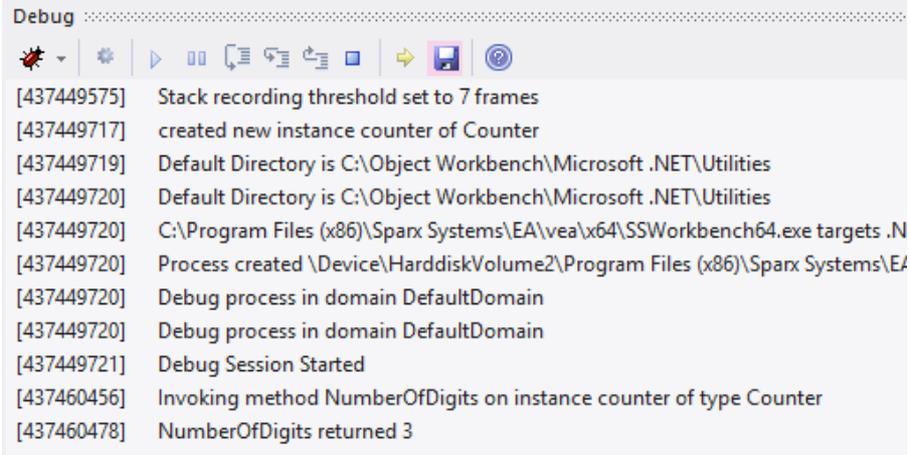
Accéder

Dans la fenêtre Établi , cliquez-droit sur l'instance sur laquelle exécuter une méthode, et sélectionnez 'Invoquer'.

Ruban	Exécuter > Outils > Tester > Ouvrir Object Établi
-------	---

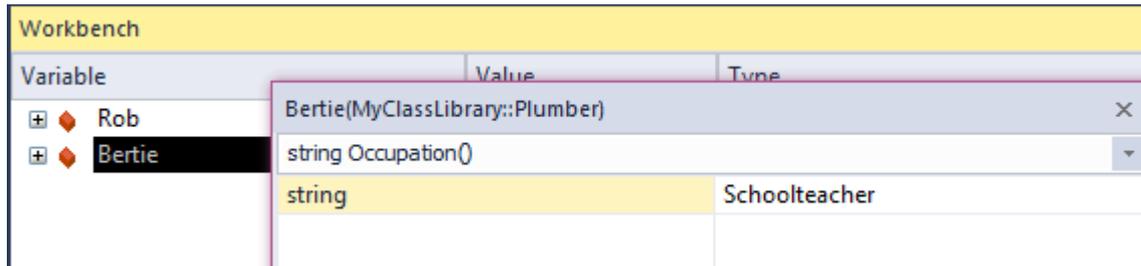
Actes

Action	Détails
Choisissez la méthode	<p>Sélectionnez une méthode dans la liste et cliquez sur le bouton Invoquer.</p> <p>Note que toutes les méthodes répertoriées sont publiques ; les méthodes privées ne sont pas disponibles.</p> 
Fournir des arguments	<p>Dans cette image, la méthode à invoquer prend un tableau d'objets comme seul argument. Vous construisez cet argument en nommant les autres instances de votre Établi que vous souhaitez transmettre à la méthode.</p> 
Types d'arguments	<p>Voici les types de paramètres pris en charge par Établi :</p> <ul style="list-style-type: none"> • Cordes

	<ul style="list-style-type: none"> • Nombres • Objets • Tableaux String • Tableaux Object
Syntaxe des arguments	<ul style="list-style-type: none"> • Chaînes comme arguments - Entourez les chaînes de guillemets si nécessaire ; par exemple, pour éviter les conflits avec les noms object Établi • Tableaux String comme arguments - Entrez les éléments qui composent le tableau, séparés par des virgules ; par exemple : « Un livre de mathématiques », « Un livre de géographie », « Un livre d'informatique » • Objets comme arguments - Type le nom object Établi comme argument ; le débogueur vérifie tout nom saisi dans un argument par rapport à sa liste d'instances Établi et remplacera cette instance dans l'appel réel à la méthode • Tableaux Object comme arguments - Entrez les noms des objets Établi pour satisfaire l'argument, séparés par des virgules : Tom, John, Peter
Invoquer	<p>Cliquez sur le bouton Invoke pour exécuter la méthode. Une sortie confirmant cette action s'affiche dans la fenêtre Débogueur .</p>  <pre> Debug [437449575] Stack recording threshold set to 7 frames [437449717] created new instance counter of Counter [437449719] Default Directory is C:\Object Workbench\Microsoft .NET\Utilities [437449720] Default Directory is C:\Object Workbench\Microsoft .NET\Utilities [437449720] C:\Program Files (x86)\Sparx Systems\EA\vea\x64\SSWorkbench64.exe targets .N [437449720] Process created \Device\HarddiskVolume2\Program Files (x86)\Sparx Systems\EA/ [437449720] Debug process in domain DefaultDomain [437449720] Debug process in domain DefaultDomain [437449721] Debug Session Started [437460456] Invoking method NumberOfDigits on instance counter of type Counter [437460478] NumberOfDigits returned 3 </pre>

Définition Propriétés

Pour les langages qui supportent les propriétés, nous pouvons définir la valeur d'une propriété d'un Object de la même manière que nous invoquons une méthode. Sélectionnez l'instance dans Établi et utilisez son menu contextuel pour sélectionner l'option « Invoquer ». Vous trouverez les propriétés exposées par la classe classées par ordre alphabétique, ainsi que ses méthodes. Vous serez invité à fournir la nouvelle valeur de la propriété. Type la valeur comme vous l'auriez saisie pour le paramètre dans un appel de méthode. Cette image montre la modification de la propriété *Occupation* d'une personne appelée *Bertie* ; *Bertie* étant un type de *personne*.



Débogage et Établi

Pendant que vous travaillez dans l'Établi, vous souhaitez peut-être déboguer une ou plusieurs méthodes que vous développez ou étudiez. Cela peut être facilement accompli. Les mêmes fonctionnalités de l'Analyseur d'Exécution d'Enterprise Architect sont disponibles pour les utilisateurs de l'Établi Object. Le débogage peut être effectué pendant la construction et la destruction object ainsi que pendant l'exécution d'une méthode. Pour accéder au débogueur, placez simplement un point d'arrêt aux points où vous devez parcourir le code. Vous pouvez également définir la condition sur ces points d'arrêt pour qu'ils ne s'arrêtent que dans certaines conditions.

```
20 // The NumberOfDigits static method calculates the number of digits in a string.
21 public int NumberOfDigits(string theString)
22 {
23     int count = 0;
24     for ( int i = 0; i < theString.Length; i++ )
25     {
26         if ( Char.IsDigit(theString[i]) )
27         {
28             count++;
29         }
30     }
31     m_nResult = count;
32     return m_nResult;
33 }
34 }
35 }
```

Lors du débogage, les états des objets sont inspectés à l'aide des contrôles du débogueur. Ici, nous utilisons la fenêtre Locals pour examiner l'état de notre object pendant que l'exécution est interrompue.

Variable	Value	Type
this		Utilities.Counter
m_nResult	3	int
theString	"123"	String
i	0	int
CSS1\$0000	0	int
CSS4\$0001	false	Boolean
count	0	int

Lorsque le programme reprend, l'Object sur l'Établi reflétera tous les changements d'état.

Variable	Value	Type
counter		Utilities.Counter
m_nResult	9	int

Enregistrement et l'Établi

Pendant que vous travaillez dans Établi, vous souhaitez peut-être produire un diagramme Séquence pour une ou plusieurs méthodes que vous développez ou étudiez. Cela peut être facilement réalisé. Les mêmes fonctionnalités de l'Analyseur d'Exécution d'Enterprise Architect sont disponibles dans l'Object Établi. Vous pouvez même commencer une session Établi en enregistrant d'abord un diagramme Séquence, comme moyen de visualiser ce sur quoi vous prévoyez de travailler.

Définir le Marqueur Enregistrement

```

134
135
136
137
138
139
140
141
142
143
...
public bool Test(string name)
{
    Person px = Find(name);
    if(px != null)
    {
        return true;
    }
    return IsFriend(name);
}

```

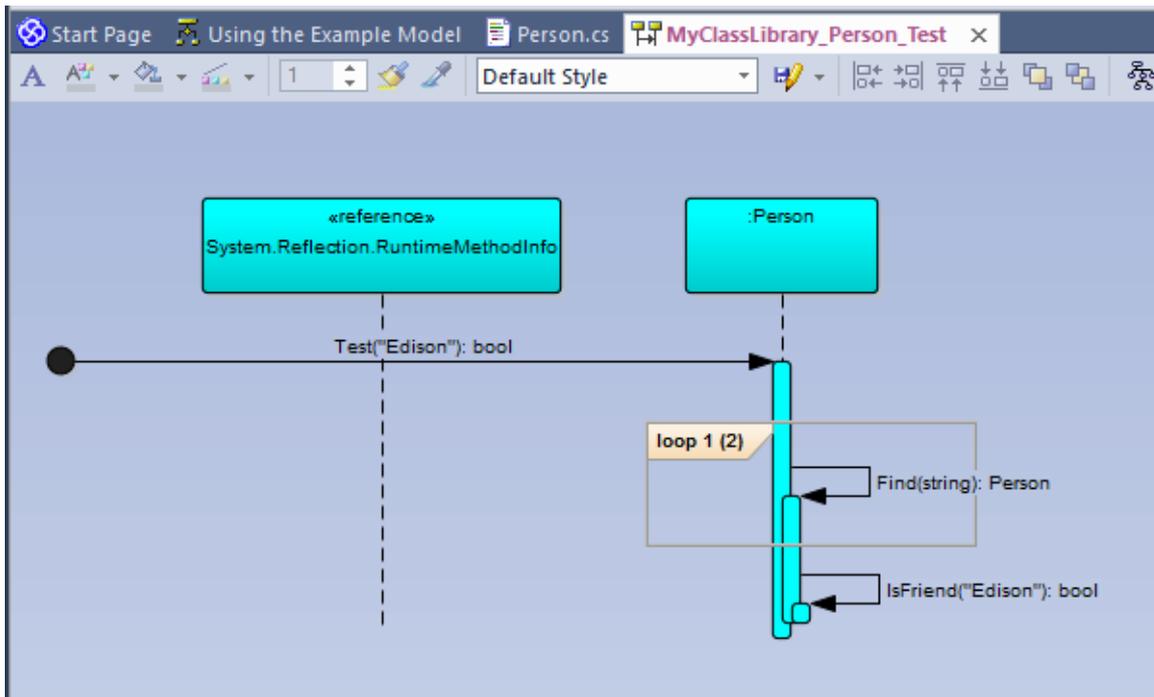
Invoquer la méthode

Workbench		
Variable	Value	Type
John		MyClassLibrary.CPerson
Robert		MyClassLibrary.CPerson
Sheila		MyClassLibrary.CPerson
m_Int	0	Sheila(MyClassLibrary::CPerson)
m_Float	123	int AddFriends(MyClassLibrary.CPerson[])
m_Double	1234	
AverageAge	0.00	MyClassLibrary.CPerson[] John,Robert
FriendCount	0	
Age	0	
Friends		

Vue de l' Historique d'Enregistrement

Record & Analyze				
Sequence	Instance	Method	Direction	Method
00000001		System.Reflection.RuntimeMethodIn...	Call	MyClassLibrary.Person.Test
00000002		MyClassLibrary.Person.Test	Call	MyClassLibrary.Person.Find
00000003		MyClassLibrary.Person.Test	Call	MyClassLibrary.Person.Find
00000004			Return	MyClassLibrary.Person.Test
00000005		MyClassLibrary.Person.Test	Call	MyClassLibrary.Person.IsFriend
00000006			Return	MyClassLibrary.Person.Test

Générer le Diagramme de Séquence



Suppression d'objets

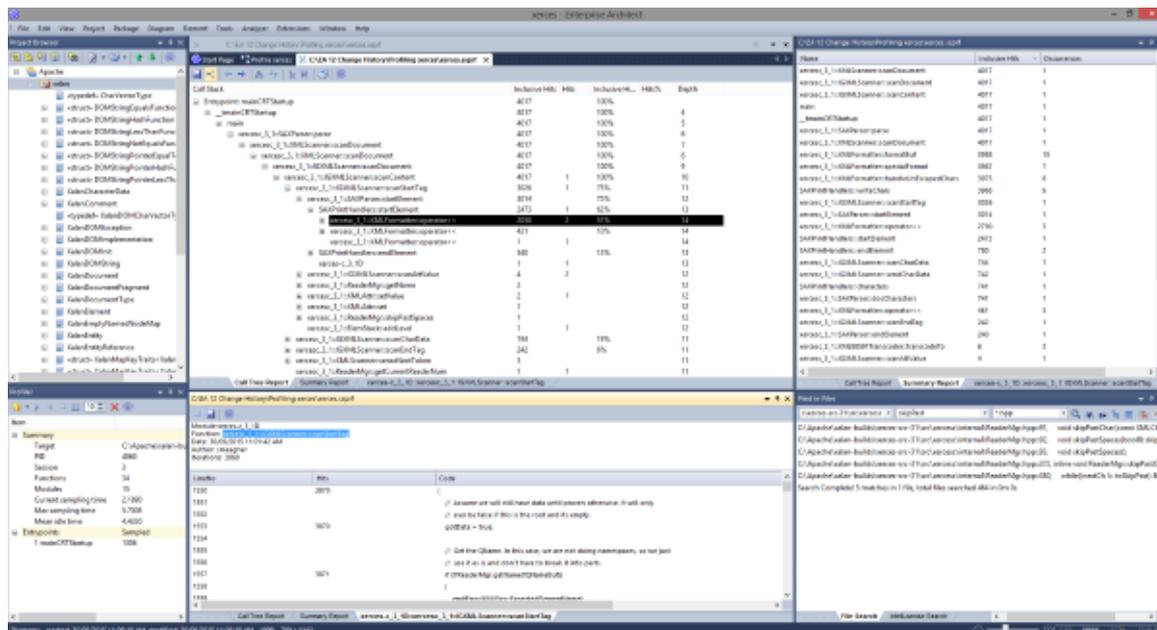
Vous pouvez facilement supprimer un objet en le sélectionnant dans l'Établi, en faisant un clic droit dessus et en sélectionnant l'option « Supprimer ».

Fermeture de l' Établi

Pour arrêter l' Établi effectuez l'une de ces actions :

- Choisissez « Réinitialiser » dans le menu contextuel de Object Établi
- Appuyez sur le bouton Arrêter de n'importe quelle barre d'outils du débogueur
- Supprimer tous les objets sur l' Établi

Profilage

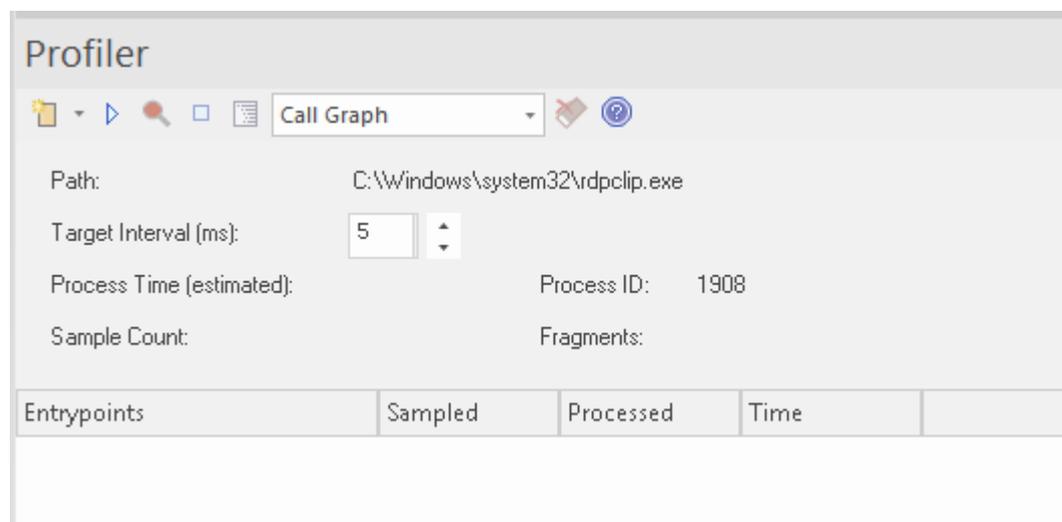


Au cours de la vie des applications logicielles, il n'est pas rare d'examiner des tâches d'application qui s'exécutent plus lentement que prévu. Vous pouvez également simplement vouloir savoir ce qui se passe lorsque vous appuyez sur ce bouton ! Vous pouvez résoudre ce problème assez rapidement dans Enterprise Architect en utilisant son Profiler. Les résultats peuvent généralement être produits en quelques secondes et vous pourrez rapidement voir les actions qui consomment l'application et les fonctions impliquées. Dans l'Analyseur d'Exécution, la fonctionnalité utilise deux stratégies distinctes : l'échantillonnage de processus et le hooking de processus. Dans l'une, des échantillons sont prélevés à intervalles réguliers pour identifier motifs gourmands en ressources CPU, tandis que dans l'autre, le processus est accroché pour enregistrer les demandes faites sur la mémoire. Les données sont analysées pour produire un Graphique d'Appel pondéré. Les comportements sont généralement identifiables comme des nœuds racines (points d'entrée) dans le graphique, ou des branches proches de ces points. Tous les rapports peuvent être examinés à la demande. Ils peuvent être enregistrés dans un fichier au sein du modèle, à la fois comme éléments d'artefact et comme messages Bibliothèque d'Équipe.

Accéder

Ruban	Exécuter > Outils > Profiler
Autre	Barre d'outils Analyseur d'Exécution : Analyseur Windows Profileur

Échantillonnage d'appel



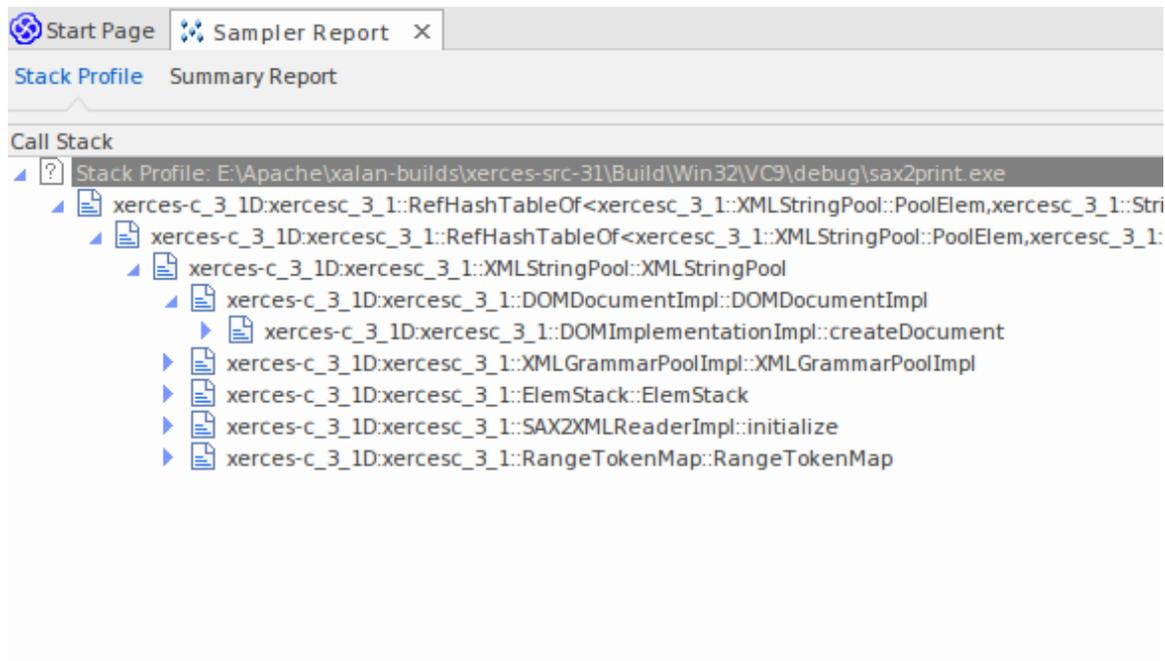
Le profileur est contrôlé à l'aide des boutons de sa barre d'outils. Ici, vous pouvez attacher le profileur à un processus existant (ou à une JVM) ou lancer l'application pour le script Analyzer actif. La fenêtre du profileur affiche les détails du processus cible au fur et à mesure de son profilage. Ces détails fournissent des informations en retour, vous permettant de voir le nombre d'échantillons prélevés. Vous disposez également d'options pour suspendre et reprendre la capture, effacer les données capturées et générer des rapports. Vous pouvez accéder à la fonctionnalité de création de rapports en mettant la capture en pause - la fonctionnalité de création de rapports est désactivée pendant que la capture de données est en cours.

Graphique d'Appel pondéré

Call Stack	Inclusive Hits	Hits
[-] xercesc_3_1::SAX2XMLReaderImpl::parse	16051	
[-] xercesc_3_1::XMLScanner::scanDocument	16051	
[-] xercesc_3_1::IGXMLScanner::scanDocument	16051	
[-] xercesc_3_1::IGXMLScanner::scanContent	16051	
[-] xercesc_3_1::IGXMLScanner::scanStartTagNS	16051	
[-] xercesc_3_1::IGXMLScanner::resolveSchemaGrammar	16051	
[-] xercesc_3_1::SchemaValidator::preContentValidation	16049	
[-] xercesc_3_1::ComplexTypeInfo::checkUniqueParticleAttribution	16049	
[-] xercesc_3_1::ComplexTypeInfo::makeContentModel	16049	
[-] xercesc_3_1::DFACContentModel::DFACContentModel	16047	
[-] xercesc_3_1::DFACContentModel::buildDFA	15998	515
[-] xercesc_3_1::CMStateSet::operator =	8174	8093
memcpy	32	32
[-] xercesc_3_1::CMStateSet::allocateChunk	27	1
_security_check_cookie	21	21
TrailUpVec	1	1
[-] xercesc_3_1::CMStateSet::~CMStateSet	3573	4
[-] xercesc_3_1::XMemory::operator delete	841	2
xerces-c_3_1D	4416	2
xercesc_3_1::CMStateSet::getBit	1036	1036
[-] xercesc_3_1::DFACContentModel::buildSyntaxTree	528	3
[-] xercesc_3_1::CMStateSet::CMStateSet	373	3
xercesc_3_1::CMStateSet::getBitCountInRange	285	285
[-] xercesc_3_1::XMemory::operator new	211	2
[-] xercesc_3_1::CMStateSet::zeroBits	154	
[-] xercesc_3_1::CMStateSetEnumerator::nextElement	153	136
[-] xercesc_3_1::RefHashTableOf<xercesc_3_1::XMLInteger,>	59	2
[-] xercesc_3_1::RefHashTableOf<xercesc_3_1::XMLInteger,>	28	2
[-] xercesc_3_1::RefHashTableOf<xercesc_3_1::XMLInteger,>	25	
[-] xercesc_3_1::DFACContentModel::makeDefStateList	25	2

Ce rapport détaillé présente l'ensemble unique de piles d'appels/comportements sous forme Graphique d'Appel pondéré. Le poids de chaque branche est représenté par un nombre de hits, qui correspond au nombre total de hits de cette branche plus tous les branches à partir de ce point. En suivant la piste de hits, vous pouvez rapidement identifier les zones de code qui ont le plus occupé le programme pendant la période de capture.

Profil de Pile



Les profils de pile permettent de découvrir les différentes manières (piles) et le nombre de façons dont une fonction particulière est invoquée pendant l'exécution du programme. Contrairement aux autres modes de profilage, ce profil est activé via l'utilisation d'un point de profil, qui est un type spécial de marqueur de point d'arrêt. Le marqueur est défini dans le code source comme tout autre point d'arrêt. Lorsque le point d'arrêt est rencontré par le programme, la pile est capturée. Lorsque vous produisez ultérieurement le rapport, les piles sont analysées et un graphique d'appels pondéré est produit. Le graphique montre les piles uniques impliquées dans cette fonction pendant la période d'exécution du profileur. La colonne « Hit Count » indique le nombre de fois que la même pile s'est produite.

```

106
107 template <class TVal, class THasher>
108 void RefHashTableOf<TVal, THasher>::initialize(const XMLSize_t modulus)
109 {
110     if (modulus == 0)
111         ThrowXMLwithMemMgr(IllegalArgumentException, XMLExcepts::HshTbl_ZeroMo
112
113     // Allocate the bucket list and zero them
114     fBucketList = (RefHashTableBucketElem<TVal>**) fMemoryManager->allocate
115     (
116         fHashModulus * sizeof(RefHashTableBucketElem<TVal>*)
117     );
118     for (XMLSize_t index = 0; index < fHashModulus; index++)
119         fBucketList[index] = 0;
120 }
121

```

Profils de Mémoire

	Heap	Virtual
Allocations:	4377	2
Frees:	0	0
Frees not	0	0
Stack holdings:	4375	2
Memory holdings	1	0

```
Module loaded:api-ms-win-core-datetime-l1-1-1, base:7e9d0000
Module loaded:api-ms-win-core-localization-obsolete-l1-2-0, base:7e71
Build version:3.6, status:0x0003, process heap:0x00110000
Auditor installed.
Module loaded:api-ms-win-appmodel-runtime-l1-1-2, base:7e6b0000
Target process ended.
Audit ended.
```

Le profil de mémoire suit les allocations, en ignorant le moment où la mémoire est libérée. Il utilise ces informations pour évaluer les demandes de mémoire du code en cours d'exécution, non pas en termes de quantité de mémoire, mais de fréquence des demandes. Le chiffre *Allocations* correspond au nombre total d'allocations de mémoire demandées. Le nombre de *traces de pile* correspond au nombre de traces de pile prises à ces moments-là, et le chiffre *Heap Holding* correspond à la quantité totale de mémoire obtenue par ces appels. Note que le profilage peut être activé et désactivé à la demande. Il n'est pas non plus nécessaire de reconstruire votre programme pour le faire fonctionner car aucun lien n'est impliqué.

Graphique de Mémoire

Call Stack	Instances	Bytes
E:\Apache\xalan-builds\xerces-src-31\Build\Win32\VC9\debug\DOMPrint.exe C:\test\materials\portrait.xml	0	0
ntdll:RtlAllocateHeap	7,068	4,830,947
ntdll:RtlpNtSetValueKey	7,068	4,830,947
ntdll:RtlDestroyMemoryBlockLookaside	7,068	4,830,947
ntdll:RtlAllocateHeap	7,068	4,830,947
ntdll	7,068	4,830,947
msvcr90d:malloc_base	4,813	3,985,885
msvcr90d:malloc_dbg	4,813	3,985,885
msvcr90d:malloc_dbg	4,813	3,985,885
msvcr90d:malloc_dbg	4,813	3,985,885
msvcr90d:malloc	4,812	3,985,734
msvcr90d:operator new	4,812	3,985,734
xerces-c_3_1d:xercesc_3_1::MemoryManagerImpl::allocate	4,807	3,985,526
xerces-c_3_1d:xercesc_3_1::RangeToken::expand	803	396,984
xerces-c_3_1d:xercesc_3_1::ValueHashTableOf<bool,xercesc_3_1::StringHasher>::put	791	37,968
xerces-c_3_1d:xercesc_3_1::XMemory::operator new	753	239,048
xerces-c_3_1d:xercesc_3_1::XMemory::operator new	333	21,652
xerces-c_3_1d:xercesc_3_1::RangeToken::doCreateMap	291	19,788
xerces-c_3_1d:xercesc_3_1::RangeToken::addRange	287	28,700
xerces-c_3_1d:xercesc_3_1::XMLString::replicate	218	12,914
xerces-c_3_1d:xercesc_3_1::RefHashTableOf<xercesc_3_1::RangeTokenElemMap,xercesc_3_1	146	7,008
xerces-c_3_1d:xercesc_3_1::RefHashTableOf<xercesc_3_1::CPMapEntry,xercesc_3_1::StringH	144	6,912
xerces-c_3_1d:xercesc_3_1::Win32TransService::Win32TransService	112	6,612
xerces-c_3_1d:xercesc_3_1::Win32TransService::Win32TransService	106	6,258

Cet exemple est un rapport produit à partir du profilage d'un programme de démonstration du projet Xerces à partir d'Apache. Le programme parcourt le Modèle Object de document (DOM) pour un fichier XML fourni.

Rapport Sommaire de Fonction

Name	Inclusive Hits
profiler/Example.Run	156
profiler/Example.main	156
java/io/FileOutputStream.write	154
java/io/PrintStream.println	154
profiler/Example.Print	154
profiler/Example.MakeItalianCars	2
profiler/Example.NewCar	2

Ce rapport récapitulatif répertorie les fonctions et uniquement celles exécutées pendant la période d'échantillonnage. Les fonctions sont répertoriées par nombre total d'appels, une fonction qui apparaît deux fois dans des piles d'appels distinctes apparaissant avant une fonction qui n'apparaît qu'une seule fois.

Rapport de Ligne de Fonction

LineNo	Hits	Code
54	1	for(int n = 0; n < 10000; n++)
55		{
56	1408	m_Cars = new Collection<Car>();
57	1408	if((n % 3) > 0)
58		{
59	938	for(int i = 0; i < 1000; i++)
60		{
61	938000	MakeItalianCars();
62		}

Ce rapport détaillé montre le code source d'une fonction ligne par ligne, affichant à côté le nombre total de fois que chacune a été exécutée. Nous avons découvert du code à l'aide de ce rapport, qui exposait des instructions case dans du code qui ne semblaient jamais être exécutées.

Support

Le profileur est pris en charge pour les programmes écrits en C, C++, Visual Basic, Java et les langages Microsoft .NET . Le profilage de la mémoire est actuellement disponible pour les programmes C et C++ natifs.

Notes

- Le profileur est disponible dans l'édition Professional Enterprise Architect et au-dessus
- Le Profiler peut également être utilisé sous WINE (Linux et Mac) pour le profilage d'applications Windows standard déployées dans un environnement WINE

Exigences du système

À l'aide du Profiler, vous pouvez analyser les applications créées pour ces plateformes :

- MicrosoftTM Native (C++, C, Visual Basic)
- Microsoft .NET (prenant en charge un mélange de code géré et non géré)
- Java

Applications natives Microsoft

Pour les applications C, C++ ou Visual Basic, le profileur nécessite que les applications soient compilées avec le compilateur MicrosoftTM Native et que pour chaque application ou module concerné, un fichier PDB soit disponible. Le profileur peut échantillonner les configurations de débogage et de publication d'une application, à condition que le fichier PDB de chaque exécutable existe et soit à jour.

Applications Microsoft .NET

Pour les applications Microsoft .NET, le Profiler nécessite que le framework Microsoft .NET approprié soit installé et que pour chaque application ou module à analyser, un fichier PDB soit disponible.

Java

Pour Java, le profileur nécessite que le JDK approprié d'Oracle soit installé.

Les classes d'intérêt doivent également avoir été compilées avec des informations de débogage. Par exemple : « `java -g *.java` »

- Une nouvelle instance de la machine virtuelle d'application est lancée à partir d' Enterprise Architect - aucune autre action n'est requise
- La machine virtuelle d'application existante est attachée à partir d' Enterprise Architect - la Virtual Machine Java cible doit avoir été lancée avec l'agent de profilage Enterprise Architect

Voici des exemples de lignes de commande pour créer une machine virtuelle Java avec un agent JVMTI spécifique :

1. `java.exe -cp "%classpath%;\ " -agentpath:"C:\Program Files (x86)\ Sparx Systems \EA\vea\x86\ssamplerlib32" monapplication`
2. `java.exe -cp "%classpath%;\ " -agentpath:"C:\Program Files (x86)\ Sparx Systems \EA\vea\x64\ssamplerlib64" monapplication`

(Reportez-vous à la documentation du JDK pour plus de détails sur l'option de démarrage de la machine virtuelle -agentpath.)

Démarrage

Le Profiler peut être utilisé pour étudier les problèmes de performances, en vous proposant quatre outils distincts parmi lesquels choisir, à savoir :

- Graphique d'Appel
- Profil de Pile
- Profil de mémoire
- Fuites de Mémoire

Vous sélectionnez ces outils dans la barre d'outils du Profiler.

Accéder

Ruban	Exécuter > Outils > Profiler
-------	------------------------------

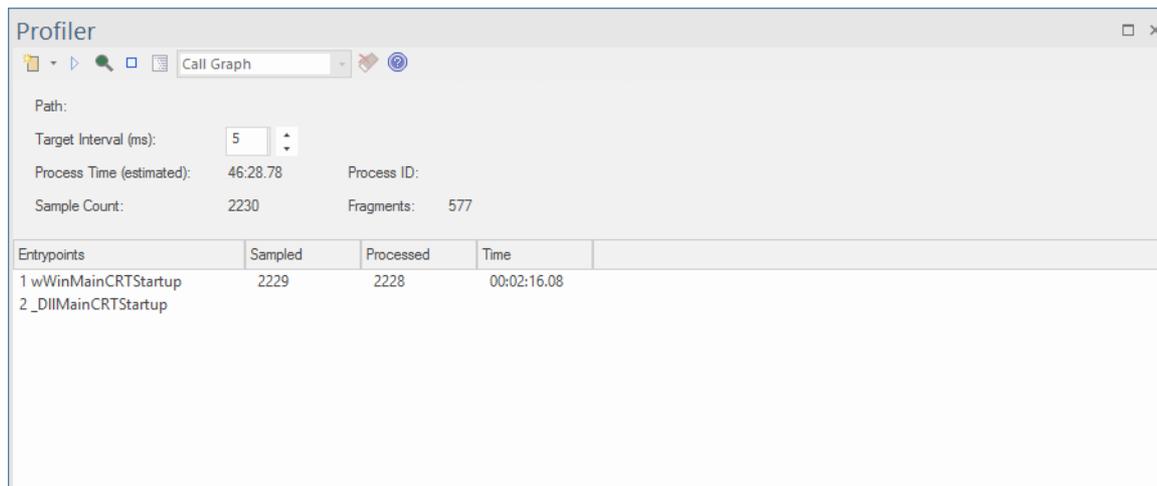
Outils

Outil	Description
Graphique d'Appel	Analyse les performances en prenant des échantillons au cours d'une activité dans un programme. Chaque échantillon représente une pile. Les échantillons sont prélevés à des intervalles contrôlés à l'aide de la barre d'outils. Dans ce scénario, les mauvaises performances sont évaluées en fonction des motifs de comportement qui se répètent le plus pendant la période d'échantillonnage. Ce chiffre est utilisé pour pondérer le Graphique d'Appel produit.
Profil de mémoire	Analyse les performances en exploitant les allocations de mémoire effectuées par un programme. Dans ce scénario, les performances médiocres sont évaluées en fonction des activités qui sollicitent le plus de mémoire. Ce chiffre est utilisé pour pondérer le Graphique d'Appel produit.
Profil de Pile	Le Stack Profiler vous permet de définir un marqueur dans votre code source afin que chaque fois que l'exécution atteint ce marqueur, une trace de pile complète soit capturée. Au fur et à mesure que l'application continue de s'exécuter et que la position marquée est accessible à partir de divers emplacements dans l'exécutable en cours d'exécution, une image très détaillée et utile est créée, montrant les points chauds et les scénarios d'utilisation pour un point particulier du code. Le rapport Profil de Pile, comme le rapport Profil de mémoire, s'affiche dans l'ordre « pile inversée ». Cela signifie que la racine du rapport est toujours un nœud unique (dans ce cas, le marqueur) et que l'arbre se déploie ensuite pour afficher tous les différents emplacements à partir desquels la position marquée a été accédée.
Fuites de Mémoire	Analyse les fuites de mémoire en récupérant les opérations de mémoire effectuées par un programme. Le résultat est un Graphique d'Appel présentant les piles d'appels qui alloué de la mémoire pour lesquelles aucune opération de libération n'a été détectée.

Boutons de la barre d'outils

Bouton	Action
	Affiche un menu d'options pour gérer votre session de profilage.
	Lance l'application configurée à profiler. Par défaut, il s'agit de l'application configurée dans le script Analyzer actif.
	Indique l'état de l'échantillonneur. Lorsqu'il est vert, l'échantillonnage est activé ; lorsqu'il est rouge, l'échantillonnage est désactivé.
	Arrête le processus du profileur ; si des échantillons ont été collectés, les boutons Rapport et Supprimer les données sont actifs.
	Génère un rapport à partir de la collecte de données actuelle.
	Affiche l'outil de profilage utilisé, qui détermine les champs affichés dans la fenêtre du profileur. Cliquez sur la flèche déroulante et sélectionnez un autre outil, ce qui modifie les champs de la fenêtre.
	Supprime les données collectées. Vous êtes invité à confirmer la suppression.
	Affiche la rubrique d'aide de cette fenêtre.

Graphique d'Appel



- Découvrez rapidement ce que fait un programme à tout moment
- Identifiez facilement les problèmes de performances
- Soyez surpris de la rapidité avec laquelle vous pouvez réaliser des améliorations
- Voyez vos améliorations au travail et ayez-en la preuve
- Support des plateformes C/C++, .NET et Java

Usage

L'option « Graphique d'Appel » est généralement utilisée dans les situations où une activité s'exécute plus lentement que prévu, mais elle peut également être utilisée simplement pour mieux comprendre les motifs de comportement en jeu au cours d'une activité.

Opération

Le Profiler fonctionne en prenant des échantillons (ou piles d'appels) à intervalles réguliers sur une période donnée ; l'intervalle est défini à l'aide de la barre d'outils du Profiler. Vous pouvez utiliser le Profiler pour exécuter un programme particulier ou vous pouvez vous connecter à un processus existant. La capture du Profiler est contrôlée et vous pouvez suspendre et reprendre la capture à tout moment. Vous pouvez également choisir de lancer la capture immédiatement au démarrage du Profiler. Si nécessaire, vous pouvez supprimer les échantillons capturés et recommencer au cours de la même session. Si vous ne pouvez pas continuer avec la même session, le redémarrage du Profiler est rapide et facile.

Note que le champ « Temps de processus (estimé) » affiche une estimation de la durée d'exécution du processus en cours de profilage, en tenant compte des interruptions du processus par le profileur lors de la collecte des échantillons.

Résultats

Les résultats peuvent être produits à tout moment au cours de la session. Cependant, la capture doit être désactivée pour que le bouton Rapport devienne actif. C'est à vous de décider combien de temps vous laissez le Profiler exécuter . Vous pouvez savoir quand une activité est terminée, ou cela peut être apparent pour d'autres raisons. La raison pour laquelle vous êtes ici peut être qu'une activité ne se termine pas du tout.

Le bouton Rapport sera activé soit en mettant la capture en pause, soit en arrêtant complètement le Profiler.

Les résultats sont affichés dans une vue Rapport . Le rapport s'ouvre avec trois onglets initialement visibles : le Graphique d'Appel , le Rapport de synthèse (Résumé des fonctions) et les onglets Analyse Hit . Les rapports peuvent être enregistrés dans un fichier, stockés dans le modèle sous forme d'Artefacts ou publiés dans la Bibliothèque d'Equipe .

L'onglet Graphique d'Appel

	Inclusive Hits	UFP	Hits	Inclusive Hits%	Hits%
EA:CNIEMSchemaImporterDlg::OnBnClickedImport	1,283	1		54%	
EA:CNIEMSchemaImporter::ImportSchemas	862	1		36%	
EA:CNIEMNamespaceCreator::CreateNIEMNamespace	398	1		17%	
EA:CNIEMNamespaceCreator::CreateSchemaTypeProperties	259	1		11%	
EA:CNIEMNamespaceCreator::CreateComplexTypeProperties	147	1		6%	
EA:CNIEMNamespaceCreator::CreateNIEMAttribute	109	35		5%	
EA:CDaoDataMan::UpdateEx	109	43		5%	
EA:CDaoDataMan::UpdateAutoCounter	76	32		3%	
EA:CSSRecordset::Update	76	46		3%	
EA:CSSRecordset::Update	15	7		1%	
EA:SACCommand::SACCommand	15	13		1%	
EA:SACCommand::setCommandText	15	14		1%	
EA:SACCommand::ParseCmd	15	14		1%	
EA:SACCommand::ParseInputMarkers	11	10			
EA:SACCommand::CreateParam	8	7			
EA:sParams::find	8	7			
EA:SACCommand::CompareIdentifier	8	7			
EA:SACString::CompareNoCase	4	4			
ucrtbased	4	4			
EA:SACString::CompareNoCase	3	2			
EA:SACString::CompareNoCase	1	1	1		
EA:SACCommand::CreateParam	1	1			
EA:SACCommand::CreateParam	1	1			

L'onglet Rapport récapitulatif

Functions	Inclusive Hits	Depth	Modules	Occurrences
invoke_main	2392	4	EA	1
wWinMain	2392	5	EA	1
__scr_t_common_main	2392	2	EA	1
__scr_t_common_main_seh	2392	3	EA	1
CBCGPDIALOG::DoModal	1771	80	EA	2
CMainFrame::WindowProc	1671	103	EA	9
CBCGPFrameWnd::OnCommand	1625	21	EA	1
CMainFrame::OnCmdMsg	1625	24	EA	1
CMainFrame::OnImportNIEMXSD	1625	27	EA	1
CSSDialog::DoModal	1622	28	EA	1
CBCGPDIALOG::PreTranslateMessage	1538	92	EA	3
CSSDialog::PreTranslateMessage	1535	40	EA	2
CBCGPButton::OnLButtonUp	1533	105	EA	2
CBCGPDIALOG::OnCommand	1533	123	EA	2
CNIEMSchemaImporterDlg::OnBnClickedImport	1283	77	EA	1
CNIEMSchemaImporter::ImportSchemas	862	78	EA	1
CNIEMNamespaceCreator::CreateNIEMNamespace	398	79	EA	1
CDaoDataMan::UpdateEx	398	86	EA	20
CSSRecordset::Update	322	89	EA	23
CNIEMSchemaImporter::ImportSchemas	304	78	EA	1

L'onglet Analyse Hit

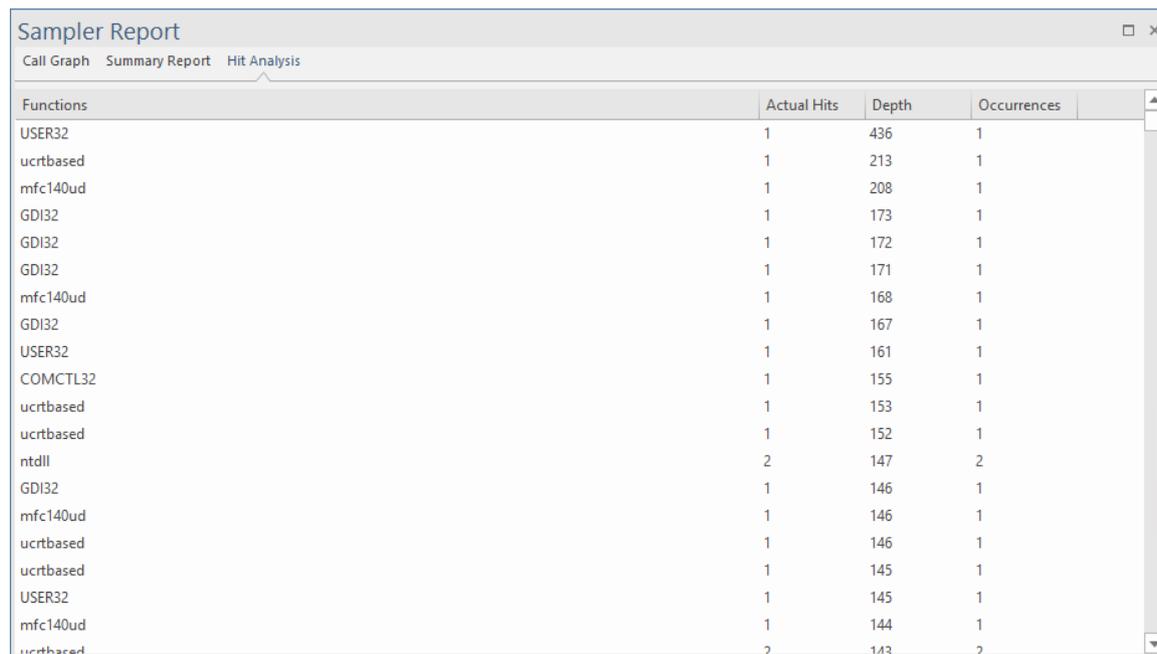
L'onglet « Analyse Hit » affiche un certain nombre de colonnes :

- Fonction : le nom de la fonction (ou du module s'il n'y a pas de symbole pour le module)
- Hits : le nombre d'échantillons prélevés, dans lesquels la fonction était en cours d'exécution.
- Profondeur : le numéro de l'image ou la profondeur de la pile à laquelle le coup a eu lieu.

- Occurrences : le nombre de fois où la fonction a été touchée à cette profondeur de pile particulière

Le nombre de hits sur une fonction particulière est agrégé en fonction de la profondeur de la trame de pile lors de l'échantillonnage.

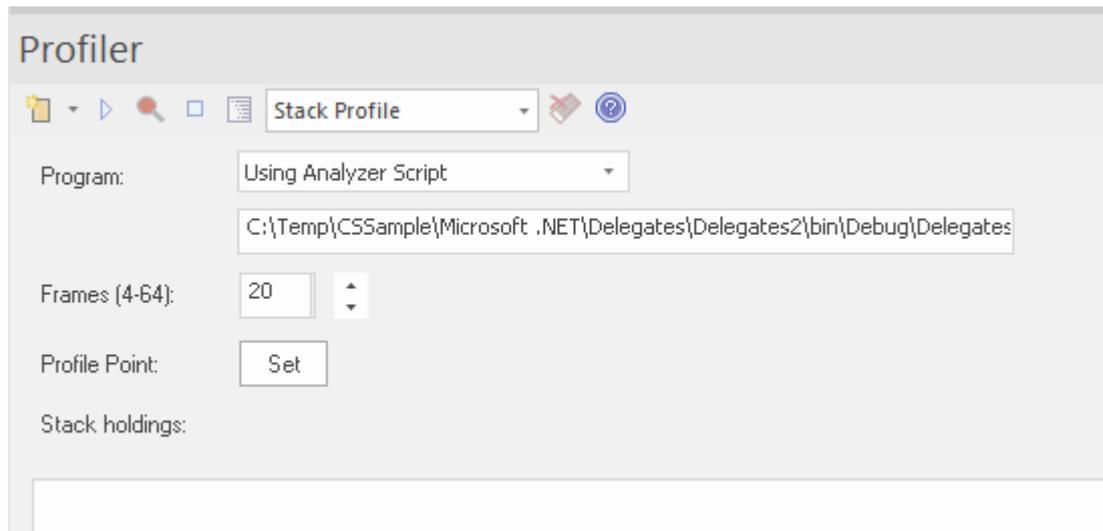
Si le nom de la fonction n'est pas disponible, par exemple les DLL système Windows telles que User32 ou les DLL sans informations de débogage, le nom du module est affiché à la place.



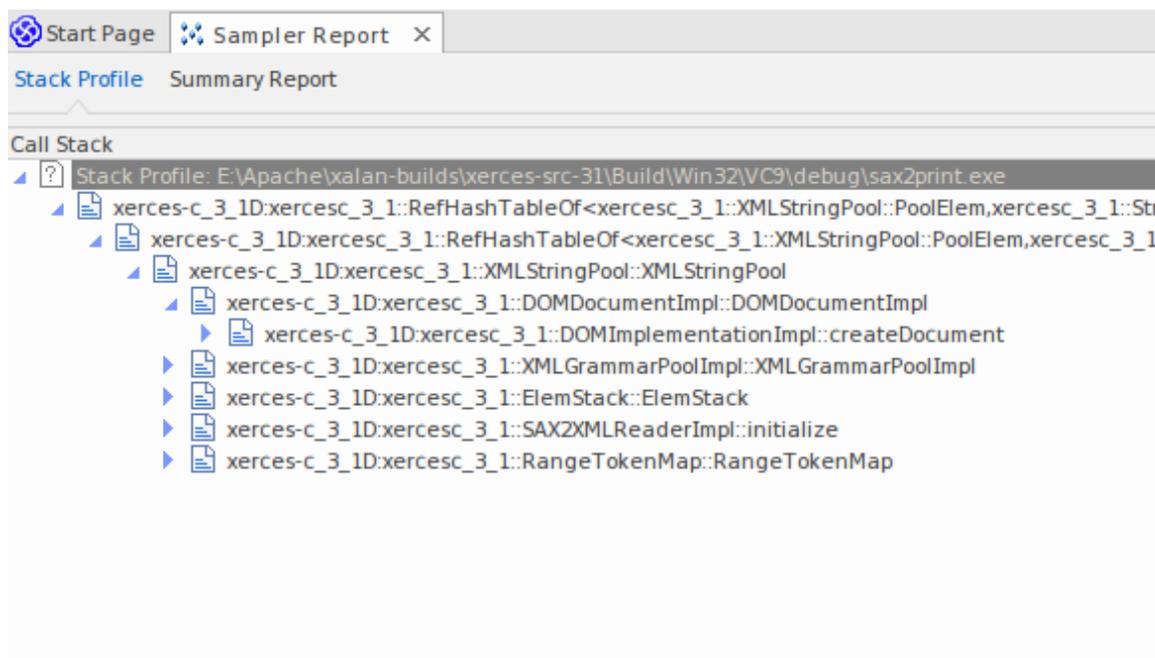
Functions	Actual Hits	Depth	Occurrences
USER32	1	436	1
ucrtbased	1	213	1
mfc140ud	1	208	1
GDI32	1	173	1
GDI32	1	172	1
GDI32	1	171	1
mfc140ud	1	168	1
GDI32	1	167	1
USER32	1	161	1
COMCTL32	1	155	1
ucrtbased	1	153	1
ucrtbased	1	152	1
ntdll	2	147	2
GDI32	1	146	1
mfc140ud	1	146	1
ucrtbased	1	146	1
ucrtbased	1	145	1
USER32	1	145	1
mfc140ud	1	144	1
ucrtbased	2	143	2

Profil de Pile

Le Stack Profiler vous permet de définir un marqueur dans votre code source afin que chaque fois que l'exécution atteint ce marqueur, une trace de pile complète soit capturée. Au fur et à mesure que l'application continue de s'exécuter et que la position marquée est accessible à partir de divers emplacements dans l'exécutable en cours d'exécution, une image très détaillée et utile est créée, montrant les points chauds et les scénarios d'utilisation pour un point particulier du code.



Le rapport Profil de Pile, comme le rapport Profil de mémoire, s'affiche dans l'ordre « pile inversée ». Cela signifie que la racine du rapport est toujours un nœud unique (dans ce cas, le marqueur) et que l'arbre se déploie ensuite pour afficher tous les différents emplacements à partir desquels la position marquée a été accédée.



Usage

Utilisez le mode Profil de Pile pour générer un rapport qui montre les différentes manières dont une fonction peut être invoquée pendant l'exécution d'un programme. Déterminez les parties du modèle qui s'appuient sur cette fonction et leur fréquence.

Opération

```
106
107 template <class TVal, class THasher>
108 void RefHashTableOf<TVal, THasher>::initialize(const XMLSize_t modulus)
109 {
110     if (modulus == 0)
111         ThrowXMLwithMemMgr(IllegalArgumentException, XMLExcepts::HshTbl_ZeroMo
112
113     // Allocate the bucket list and zero them
114     fBucketList = (RefHashTableBucketElem<TVal>**) fMemoryManager->allocate
115     (
116         fHashModulus * sizeof(RefHashTableBucketElem<TVal>*)
117     );
118     for (XMLSize_t index = 0; index < fHashModulus; index++)
119         fBucketList[index] = 0;
120 }
121
```

Les modes de profilage sont sélectionnés à l'aide de la barre d'outils du contrôle Profiler. Si un point de profilage est déjà créé, il est affiché. Le point de profilage est le point auquel les traces de pile sont capturées. Vous pouvez définir le point de profilage à l'aide du bouton Définir sur le contrôle lui-même, une fois le mode sélectionné. Après avoir choisi le point de profilage, générez le projet pour vous assurer que tout est à jour, puis démarrez le profileur. Le nombre de holdings de pile uniques détectés est visible pendant l'exécution.

Résultats

Un résultat peut être généré en cliquant sur le bouton Rapport dans la barre d'outils du contrôle du profileur. Ce bouton est activé lorsque :

- La capture est désactivée (à l'aide du bouton Pause) ou
- Le profileur est arrêté (en utilisant le bouton Stop)

Les résultats produits sont affichés sous forme de graphique d'appels pondérés, où les lignes du graphique représentent une pile unique et sont pondérées pour afficher en premier les piles à fréquence plus élevée. Le rapport peut ensuite être enregistré, soit dans un fichier, soit dans le modèle, à l'aide du menu contextuel du rapport lui-même.

Profil de mémoire

The Profiler window displays a Memory Profile for the program 'Using Analyzer Script' running at 'C:\Code Samples\VEA\Microsoft Native\CityLoop\debug'. The profile shows 16 frames over an elapsed time of 3.3 seconds. The summary table is as follows:

	Heap	Virtual
Allocations:	4377	2
Frees:	0	0
Frees not	0	0
Stack holdings:	4375	2
Memory holdings	1	0

Below the summary, the following events are listed:

- Module loaded: api-ms-win-core-datetime-l1-1-1, base: 7e9d0000
- Module loaded: api-ms-win-core-localization-obsolete-l1-2-0, base: 7e71
- Build version: 3.6, status: 0x0003, process heap: 0x00110000
- Auditor installed.
- Module loaded: api-ms-win-appmodel-runtime-l1-1-2, base: 7e6b0000
- Target process ended.
- Audit ended.

The Call Stack window shows the following entries:

Call Stack	Instances	Bytes	Line
ntdll:RtlAllocateHeap	4,375	1,237,419	-1
user32:PostMessageA ? (+0x00BD, +189)	913	233,728	-1
user32:ShutdownBlockReasonDestroy ? (+0x0A6A, +2666)	632	40,448	-1
ucrtbased:toupper	541	100,374	-1
user32:GetClientRect ? (+0x0103, +259)	480	245,760	-1
gdi32:SetAbortProc ? (+0x032F, +815)	193	17,756	-1
gdi32>CreateFontIndirectExW ? (+0x0051, +81)	145	13,340	-1
winex11:SetFocus ? (+0x0A79, +2681)	121	4,992	-1
ntdll:LdrGetDllHandle ? (+0x0072, +114)	88	54,628	-1
ntdll:RtlDosPathNameToNtPathName_U_WithStatus ? (+0x0300, +768)	74	14,864	-1
gdi32>SelectObject ? (+0x00E2, +226)	58	33,456	-1
user32:GetTitleBarInfo ? (+0x09AF, +2479)	56	16,296	-1
user32:ShutdownBlockReasonDestroy ? (+0x0A6A, +2666)	52	29,212	-1
gdi32:GetCharWidthInfo ? (+0x0461, +1121)	52	832	-1

- Évaluez rapidement les performances des activités qui vous intéressent
- Rien n'influence davantage une discussion que les preuves
- Récompensez vos efforts en travaillant dans les domaines qui feront la différence

- Surprenez-vous en proposant des optimisations dont vous ignoriez peut-être l'existence

Usage

Le profil de mémoire peut être utilisé pour révéler les performances des activités en termes de consommation de mémoire. En utilisant ce mode, un utilisateur pourrait s'intéresser à la fréquence des demandes de mémoire au cours d'une tâche. Il serait moins intéressé par la quantité réelle consommée. Une activité bien gérée peut effectuer relativement peu d'appels pour allouer des ressources, mais allouer suffisamment de mémoire pour effectuer son travail efficacement. D'autres activités peuvent effectuer plusieurs milliers de demandes, ce qui les rend généralement moins efficaces. Ce mode est utile pour détecter ces scénarios.

Opération

Le profil de mémoire fonctionne en accrochant le processus en question, de sorte que le programme doit être lancé à l'aide de l'outil dans Enterprise Architect . Contrairement à l'option Graphique d'Appel , vous ne pouvez pas vous rattacher à un processus existant. Lorsque le programme est démarré, les mécanismes d'accrochage suivent l'allocation de mémoire ; ces informations sont collectées et rassemblées dans Enterprise Architect . Vous pouvez facilement surveiller le nombre d'allocations effectuées. De plus, le processus est contrôlé ; c'est-à-dire que les crochets de mémoire peuvent être activés et désactivés à la demande. Si vous avez mal synchronisé une action, vous pouvez suspendre la capture, supprimer les données et reprendre la capture facilement.

Résultats

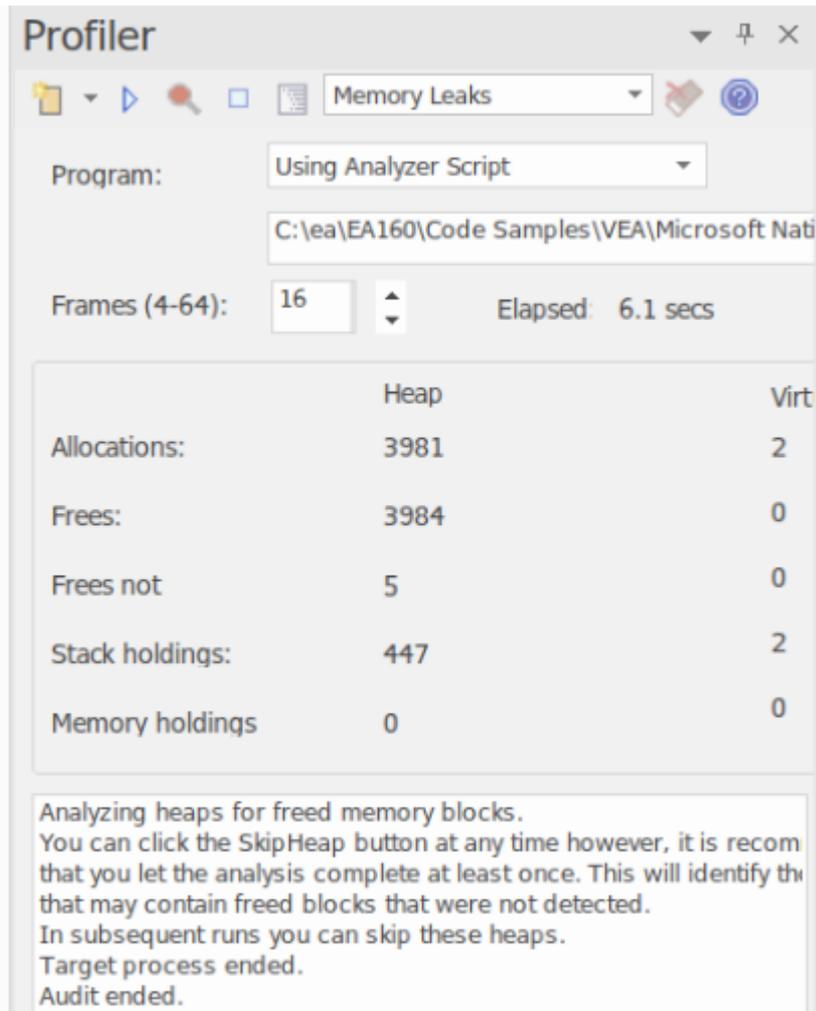
Les résultats peuvent être produits à tout moment au cours de la session. Cependant, la capture doit être désactivée pour que le bouton Rapport devienne actif. C'est vous qui décidez combien de temps vous laissez le Profiler exécuter . Vous activez le bouton Rapport en mettant en pause la capture ou en arrêtant complètement le Profiler.

Les résultats sont affichés dans une vue Rapport . Le rapport s'ouvre initialement avec deux onglets visibles : un seul Graphique d'Appel pondéré et un résumé de fonction. Le Graphique d'Appel représente toutes les piles d'appels qui ont conduit à des allocations de mémoire, qui sont agrégées et pondérées en fonction de la fréquence du motif .

Exigences

Pour obtenir les meilleurs résultats, l'image et ses modules doivent être créés avec les informations de débogage incluses et sans optimisations. Tout module doté de l'optimisation Frame Pointer Omission (FPO) est susceptible de produire des résultats trompeurs.

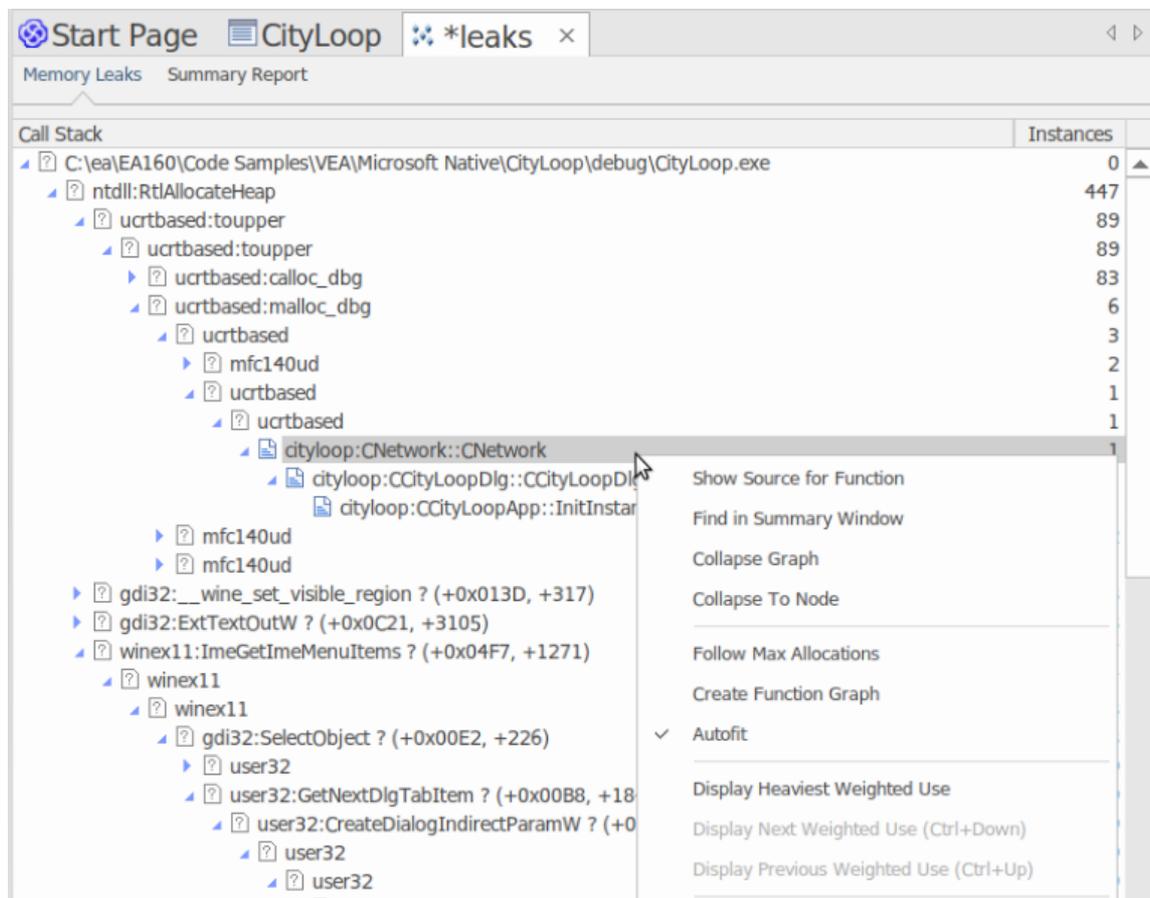
Fuites de Mémoire



	Heap	Virt
Allocations:	3981	2
Frees:	3984	0
Frees not	5	0
Stack holdings:	447	2
Memory holdings	0	0

Analyzing heaps for freed memory blocks.
You can click the SkipHeap button at any time however, it is recom that you let the analysis complete at least once. This will identify th that may contain freed blocks that were not detected.
In subsequent runs you can skip these heaps.
Target process ended.
Audit ended.

Le contrôle Profiler, affichant le nombre d'allocations de mémoire et le nombre d'opérations qui libèrent de la mémoire.



Un programme bien mené.

La détection des fuites de mémoire est une voie bien connue. Bien que de nombreuses autres options intéressantes soient disponibles, nous pensons que notre approche présente des avantages majeurs, tels que :

- Aucun changement n'a été apporté à la version existante du projet
- Aucun fichier d'en-tête requis par le code du projet
- Aucune dépendance d'exécution à prendre en compte
- Aucune configuration système à prendre en compte

Usage

Une personne peut utiliser ce mode pour suivre les fuites de mémoire dans une application ou dans une activité au sein de l'application. Une fuite de mémoire du point de vue du profileur est un appel réussi effectué vers une fonction d'allocation de mémoire qui renvoie une adresse mémoire pour laquelle aucun appel correspondant n'est effectué pour libérer cette adresse.

Opération

La détection des fuites de mémoire fonctionne par hooking. Les routines de mémoire du processus sont hookées pour suivre le moment où la mémoire est à la fois alloué et libérée. Les piles d'appels sont capturées au moment de l'allocation et ces informations sont rassemblées dans Enterprise Architect pour produire un rapport sous la forme d'un Graphique d'Appel . La capture est contrôlée ; c'est-à-dire que les mécanismes de hooking peuvent être activés ou désactivés à la demande.

Selon le type de programme et sa consommation de mémoire, vous pouvez employer une stratégie appropriée. Pour les petits programmes, vous pouvez suivre le programme du début à la fin. Pour les programmes à fenêtres plus volumineux,

il serait probablement préférable d'activer la capture avant et après une tâche spécifique pour éviter de suivre trop de données.

Résultats

Les résultats peuvent être produits à tout moment au cours de la session. Cependant, la capture doit être désactivée pour que le bouton Rapport devienne actif. C'est vous qui décidez combien de temps vous laissez le Profiler exécuter . Vous activez le bouton Rapport en mettant en pause la capture ou en arrêtant complètement le Profiler.

Les résultats sont affichés dans une vue Rapport . Le rapport s'ouvre initialement avec deux onglets visibles : un seul Graphique d'Appel pondéré et un résumé de fonction. Le Graphique d'Appel décrit toutes les piles d'appels qui ont conduit à des allocations de mémoire et sont agrégées et pondérées en fonction de la fréquence du motif .

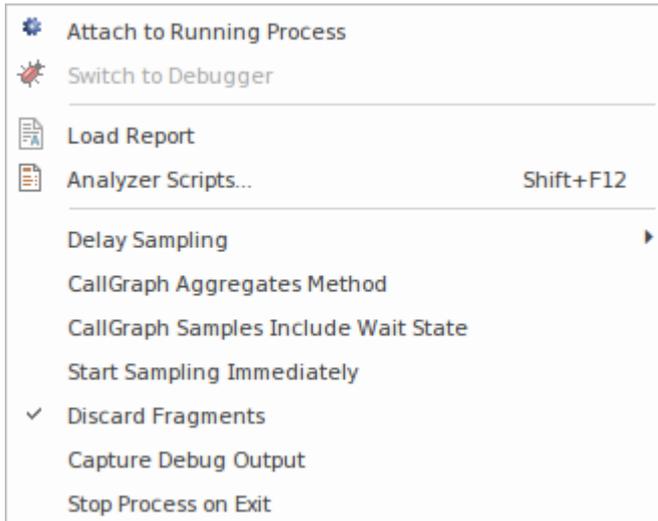
Rapports peuvent contenir une quantité variable de « bruit ». Pour vous concentrer sur un domaine qui vous préoccupe particulièrement, recherchez une fonction que vous connaissez dans le rapport récapitulatif et utilisez-la pour accéder directement à la ligne du graphique où elle est présentée.

Exigences

Pour obtenir les meilleurs résultats, l'image et ses modules doivent être créés avec les informations de débogage incluses et sans optimisations. Tout module doté de l'optimisation Frame Pointer Omission (FPO) est susceptible de produire des résultats trompeurs.

Options de Réglage

La première icône de la barre d'outils de la fenêtre Profiler affiche une liste d'options que vous pouvez définir pour personnaliser votre session de profilage.



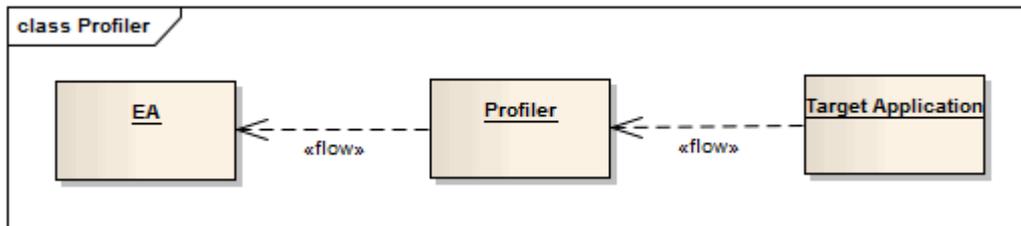
Options

Option	Description
Attacher au processus en cours d'exécution	Sélectionnez cette option pour afficher la dialogue « Attacher au processus », à partir de laquelle vous choisissez un processus actif à profiler.
Passer à Débogueur	Sélectionnez cette option pour passer des opérations de profilage à débogage. Le Débogueur dispose d'une option de menu déroulant équivalente que vous pouvez utiliser pour passer du débogage au profilage.
Rapport de charge	Sélectionnez cette option pour charger un rapport précédemment enregistré à partir du système de fichiers.
Scripts d'Analyseur	Sélectionnez cette option pour ouvrir la fenêtre Analyzer Script, qui est le référentiel modèle pour la configuration des builds, le débogage et toutes les autres options Analyseur d'Exécution Visuelle .
Échantillonnage différé	Sélectionnez cette option pour définir un délai entre le moment où vous cliquez sur l'option « Démarrer le profilage » et le début effectif du profilage. Le délai peut être de 3, 5 ou 10 secondes. Sélectionnez « Aucun » pour annuler tout délai défini.
Méthode d'agrégation CallGraph	Lorsque cette option est sélectionnée, les instances des séquences de pile identiques sont agrégées par méthode. En d'autres termes, les numéros de ligne/instructions au sein d'une méthode sont ignorés, de sorte que deux piles seront comptées comme une seule lorsqu'elles ne diffèrent que par le numéro de ligne dans leur trame finale.
Les exemples de CallGraph incluent State d'attente	Lorsque cette option est sélectionnée, le profileur échantillonne tous les threads, y compris ceux en état d'attente. Lorsqu'elle n'est pas sélectionnée, le profileur échantillonne uniquement les threads qui ont accumulé du temps CPU depuis

	l'expiration du dernier intervalle.
Démarrer l'échantillonnage immédiatement	Sélectionnez cette option pour déclencher la collecte de données immédiatement au lancement. Vous utiliserez généralement cette option pour profiler un processus au démarrage.
Jeter les fragments	<p>Lorsque les piles ne peuvent pas être rapprochées du point d'entrée d'un thread, elles sont appelées fragments. Le nombre de fragments rencontrés lors de l'échantillonnage est affiché dans la fenêtre Résumé de l'échantillonneur. Vous pouvez définir cette option pour collecter ou supprimer des fragments ; lorsque l'option Supprimer les fragments est définie sur :</p> <ul style="list-style-type: none">• Sélectionné, les fragments n'apparaissent pas dans les rapports, bien que le nombre rencontré soit toujours mis à jour• Désélectionné, une collection spéciale nommée « fragments » est créée dans le graphique d'appel pour les héberger et garantir que les données ne sont pas mélangées aux échantillons complets
Capter la sortie Déboguer	(S'applique à l'échantillonnage de processus). Lorsque cette option est sélectionnée, la sortie normalement visible pendant le débogage est capturée et affichée dans la fenêtre Déboguer . Note que seules les versions de débogage émettront généralement une sortie de débogage.
Arrêter le processus à la sortie	Cette option détermine le comportement de fin du profileur. Lorsque l'option est sélectionnée, le processus cible se termine lorsque le profileur est arrêté.

Démarrer et Arrêter le Profileur

Le profilage est un processus en deux étapes de collecte de données et de création de rapports. Dans Enterprise Architect la collecte de données présente l'avantage d'être une tâche d'arrière-plan : vous êtes donc libre de faire d'autres choses pendant l'exécution. Les informations renvoyées à Enterprise Architect sont stockées jusqu'à ce que vous génériez un rapport. Pour afficher un rapport, la capture doit être désactivée. Une fois le rapport généré, vous pouvez reprendre la capture en cliquant sur un bouton. Si, pour une raison quelconque, vous décidez de supprimer vos données et de recommencer, vous pouvez le faire facilement et sans avoir à arrêter et redémarrer le programme.



Accéder

Ruban	Exécuter > Outils > Profiler > Ouvrir le profileur
Autre	Barre d'outils Analyseur d'Exécution : Analyseur Windows Profiler

Actes

Action	Détail
Barre d'outils	
Sélection de stratégie	Sélectionnez la stratégie de profilage parmi les options disponibles dans la barre d'outils.
Démarrer le Profileur	Cliquez sur le bouton Exécuter dans la fenêtre du Profiler
Arrêter le Profileur	Le processus se termine si : <ul style="list-style-type: none"> • Vous cliquez sur le bouton Stop • L'application cible se termine, ou • Vous fermez le modèle actuel Si vous arrêtez le Profiler et que le processus est toujours en cours d'exécution, vous pouvez rapidement vous y connecter à nouveau.
Pause et Reprise de Capture	Vous pouvez mettre en pause et reprendre la capture à tout moment pendant une session. Lorsque la capture est activée, des échantillons sont collectés à partir de la cible. Lorsqu'il est en pause, le profileur entre et reste dans un état d'attente jusqu'à ce que

	la capture soit activée, que le profileur soit arrêté ou que l'application se termine.
Générer Rapports	Le bouton Rapport est désactivé pendant la capture mais est disponible lorsque la capture est désactivée.
Mode déroulant	Cliquez sur le menu déroulant et sélectionnez le mode de Profilage - Graphique d'Appel , Profil de Pile , Memory Profile ou Fuites de Mémoire .
Effacer la collecte de données	Vous pouvez effacer tous les échantillons de données collectés et reprendre à tout moment. Suspendez d'abord la capture en cliquant sur le bouton Pause. Le bouton Annuler, comme le bouton Rapport , est activé chaque fois que la capture est désactivée. En cliquant sur le bouton Annuler, il vous sera demandé de confirmer l'opération. Cette action ne peut pas être annulée.

Rapports de Ligne de Fonction

Après avoir exécuter le profileur sur une application en cours d'exécution et généré un rapport d'échantillon, vous pouvez analyser plus en détail l'activité d'une fonction spécifique répertoriée dans le rapport en générant un rapport de ligne de fonction à partir de cet élément. Un rapport de ligne de fonction indique le nombre de fois que chaque ligne de la fonction a été exécutée. Vous produisez un rapport de ligne de fonction à la fois, sur n'importe quelle méthode du rapport d'échantillon qui possède un fichier source valide. Le rapport de ligne de fonction est particulièrement utile pour les fonctions qui exécutent des boucles contenant des ramifications conditionnelles ; la couverture peut fournir une image des parties de code les plus et les moins fréquemment exécutées au sein d'une seule méthode.

Le rapport de ligne que vous générez est enregistré lorsque vous enregistrez le rapport Sampler. Le corps de la fonction est également enregistré avec le rapport de ligne de fonction pour conserver l'état de la fonction à ce moment-là.

Cette facilité ne s'applique pas aux rapports de profil mémoire.

Plateformes prises en charge

Java, Microsoft .NET et code natif Microsoft

Créer un Rapport de Ligne

Dans le rapport Sampler, cliquez-droit sur le nom de la fonction à analyser, et sélectionnez l'option 'Créer Rapport de ligne pour la fonction'.

Une fois que le profileur a lié la méthode, le rapport de ligne de fonction s'ouvre dans la fenêtre Rapport d'échantillon. Le rapport affiche le corps de la fonction, y compris les numéros de ligne et le texte. Au fur et à mesure que chaque ligne est exécutée, une valeur de hit s'accumule sur cette ligne. Un minuteur met à jour le rapport environ une fois par seconde.

LineNo	Hits	Code
21	28645	{
22	28644	if (r <= l)
23	14460	return;
24	14184	int i = l-1, j = r, p = l-1, q = r;
25		for (;;)
26		{
27	439580	while (a[++i] < a[r]) ;
28	14185	while (a[-j] > a[r])
29		if (j == l)
30		break;
31		if (j >= r)
32	14185	break;
33		
34		Exchange(a, i, j);
35		if (a[i] == a[r])
36		Exchange(a, ++p, i);
37		
38		if (a[j] == a[r])
39		Exchange(a, j, --q);
40		
41		}
42	14185	Exchange(a, i, r);
43	14185	j = i-1; i = i+1;
44	14185	for (int k = l; k < p; k++, j--)
45		

Terminer Capture du Rapport de Ligne

Une fois que suffisamment d'informations sont capturées ou que la fonction est terminée, cliquez sur le bouton Arrêter de la barre d'outils du Profiler pour arrêter l'enregistrement de la capture.

Enregistrer Rapports

Utilisez le bouton Enregistrer de la barre d'outils Pile d'Appel pour enregistrer le rapport d'échantillonneur et tous les rapports de ligne de fonction dans un fichier.

Supprimer Rapports de ligne

La fermeture de l'onglet « Rapport de ligne » fermera ce rapport, mais les données du rapport ne seront supprimées que lorsque le rapport sera enregistré.

Générer, Enregistrer et Charger des Rapports de Profil

Rapports peuvent être produits à tout moment au cours d'une session, ou naturellement à la fin d'un programme. Pour activer le bouton Rapport pendant que le programme est en cours d'exécution, vous devez cependant suspendre le profilage en activant le bouton Pause/Reprendre, ou en mettant fin au profileur avec le bouton Arrêter. Vous disposez de plusieurs options pour réviser et partager les résultats :

- Vue le rapport
- Enregistrer le rapport dans le fichier
- Distribuer le rapport comme ressource Bibliothèque d'Équipe
- Joindre le rapport en tant que document à un élément d'artefact
- Synchroniser le modèle en procédant à une rétro-ingénierie du code source ayant participé au profil

Accéder

Ruban	Exécuter > Outils > Profiler > Créer Rapport à partir des données actuelles
Profileur	Depuis la fenêtre du Profiler, cliquez sur l'icône  dans la barre d'outils.

Charger Rapport à partir du fichier

L'option est disponible dans le menu déroulant de la fenêtre du profileur

Générer Rapport

Depuis la fenêtre du Profiler, cliquez sur l'icône  dans la barre d'outils.

Rapport sur la fréquence des appels

Call Stack	Inclusive Hits	Hits
xercesc_3_1::SAX2XMLReaderImpl::parse	16051	
xercesc_3_1::XMLScanner::scanDocument	16051	
xercesc_3_1::IGXMLScanner::scanDocument	16051	
xercesc_3_1::IGXMLScanner::scanContent	16051	
xercesc_3_1::IGXMLScanner::scanStartTagNS	16051	
xercesc_3_1::IGXMLScanner::resolveSchemaGrammar	16051	
xercesc_3_1::SchemaValidator::preContentValidation	16049	
xercesc_3_1::ComplexTypeInfo::checkUniqueParticleAttribution	16049	
xercesc_3_1::ComplexTypeInfo::makeContentModel	16049	
xercesc_3_1::DFAContentModel::DFAContentModel	16047	
xercesc_3_1::DFAContentModel::buildDFA	15998	515
xercesc_3_1::CMStateSet::operator =	8174	8093
memcpy	32	32
xercesc_3_1::CMStateSet::allocateChunk	27	1
_security_check_cookie	21	21
TrailUpVec	1	1
xercesc_3_1::CMStateSet::~CMStateSet	3573	4
xercesc_3_1::XMemory::operator delete	841	2
xerces-c_3_1D	4416	2
xercesc_3_1::CMStateSet::getBit	1036	1036
xercesc_3_1::DFAContentModel::buildSyntaxTree	528	3
xercesc_3_1::CMStateSet::CMStateSet	373	3
xercesc_3_1::CMStateSet::getBitCountInRange	285	285
xercesc_3_1::XMemory::operator new	211	2
xercesc_3_1::CMStateSet::zeroBits	154	
xercesc_3_1::CMStateSetEnumerator::nextElement	153	136
xercesc_3_1::RefHashTableOf<xercesc_3_1::XMLInteger>	59	2
xercesc_3_1::RefHashTableOf<xercesc_3_1::XMLInteger>	28	2
xercesc_3_1::RefHashTableOf<xercesc_3_1::XMLInteger>	25	
xercesc_3_1::DFAContentModel::makeDefStateList	25	2

Résumé des fonctions

Name	Inclusive Hits	Occurrences
mainCRTStartup	7408	1
__tmainCRTStartup	7407	1
xercesc_3_1::XMLFormatter::handleUnEscapedChars	7351	10
xercesc_3_1::XMLFormatter::formatBuf	7351	10
xercesc_3_1::XMLFormatter::specialFormat	7351	10
SAX2PrintHandlers::writeChars	7350	10
xercesc_3_1::XMLScanner::scanDocument	7350	1
main	7350	1
xercesc_3_1::SAX2XMLReaderImpl::parse	7350	1
xercesc_3_1::XMLScanner::scanDocument	7349	1
xercesc_3_1::IGXMLScanner::scanDocument	7348	1
xercesc_3_1::XMLFormatter::formatBuf	4042	8

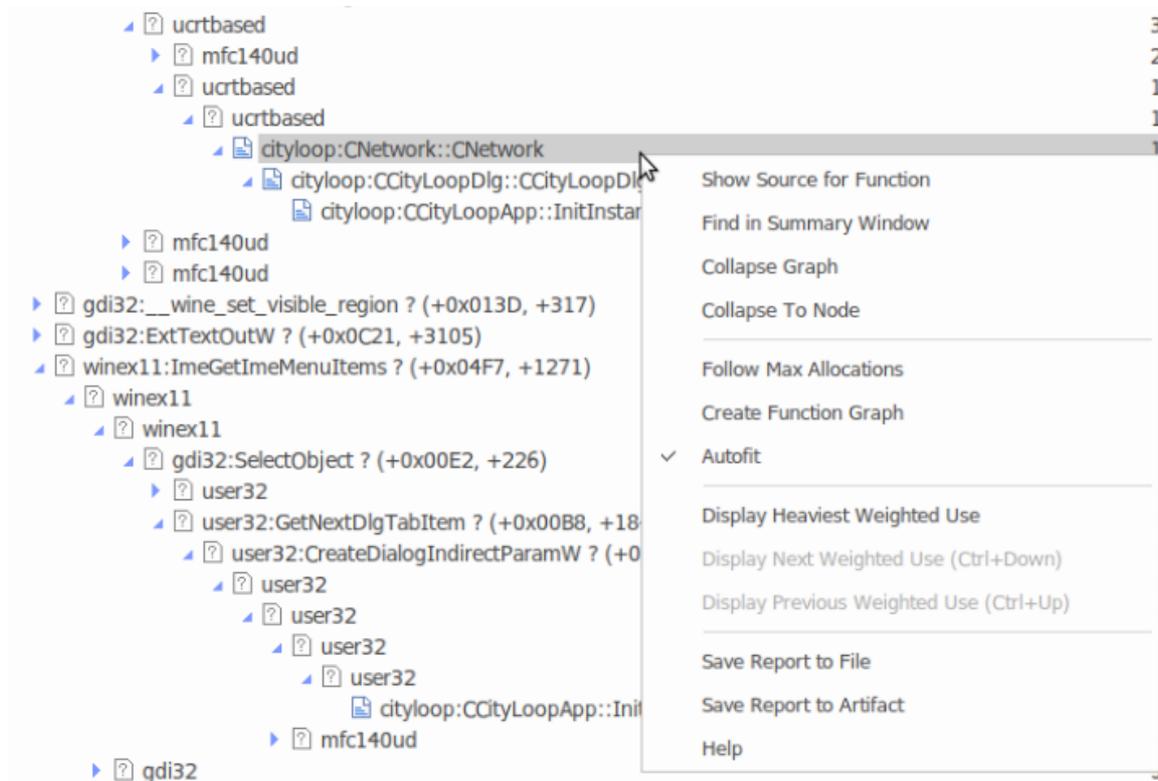
Rapport récapitulatif non filtré répertoriant toutes les fonctions participantes par ordre de succès inclusifs.

Name	Inclusive Hits	Occurrences
SAX	<input checked="" type="checkbox"/>	<input type="checkbox"/>
SAX2PrintHandlers::writeChars	7350	10
xercesc_3_1::SAX2XMLReaderImpl::parse	7350	1
xercesc_3_1::SAX2XMLReaderImpl::docCharacters	3309	2
SAX2PrintHandlers::characters	3309	2
xercesc_3_1::SAX2XMLReaderImpl::endElement	2114	1
xercesc_3_1::SAX2XMLReaderImpl::startElement	1925	1
SAX2PrintHandlers::endElement	1523	1

Vous pouvez filtrer et réorganiser les informations du rapport, de la même manière que vous le faites pour les résultats d'une recherche Modèle .

Options Rapport

Cliquez-droit sur le rapport pour afficher le menu contextuel.



Note que les options répertoriées dépendent du type de rapport affiché ; le rapport illustré ici est un rapport de profil de mémoire.

Action	Détail
Afficher la source de la fonction	Pour le cadre sélectionné, sélectionnez cette option pour afficher la ligne de code correspondante dans un éditeur de code. Les cadres dont la source est disponible sont identifiables par leur icône.
Rechercher dans la fenêtre de résumé	Sélectionnez cette option pour localiser la fonction dans la fenêtre Résumé.
Graphique de réduction	Sélectionnez cette option pour réduire l'intégralité du graphique, y compris les

	nœuds enfants, visibles ou non.
Réduire au nœud	Sélectionnez cette option pour réduire l'intégralité du graphique, puis le développer et définir le focus sur le nœud sélectionné.
Suivre les allocations maximales	Sélectionnez cette option pour développer une ligne entière dans le graphique.
Créer Rapport de ligne pour la fonction	<p>Sélectionnez cette option pour lancer le Profiler (s'il n'est pas déjà en cours d'exécution), lier immédiatement la fonction sélectionnée et la préparer pour l'enregistrement. Une fois liée, un onglet supplémentaire s'ouvre dans le Rapport Vue actuel. Ce rapport sera mis à jour instantanément, indiquant le nombre de fois que chaque ligne a été exécutée. Bien entendu, le rapport continuera d'enregistrer l'activité dans la fonction même si elle n'est pas visible.</p> <p>Notes :</p> <ul style="list-style-type: none"> • Dans les programmes fenêtrés, il est généralement nécessaire d'effectuer une action dans l'application pour provoquer l'appel de la fonction. • Cette option ne s'applique pas aux rapports de profil de mémoire
Créer un graphique de fonction	Sélectionnez cette option pour créer un onglet supplémentaire qui affiche la fonction sélectionnée de manière isolée. Pour un profil de fréquence d'appel, cela produit un graphique montrant toutes les lignes qui ont conduit à l'appel de cette fonction (c'est-à-dire les appelants). Pour un profil de mémoire, cela produit un graphique montrant toutes les lignes qui émanent de cette fonction (c'est-à-dire les appelés).
Marquer le cadre initial pour Diagramme Pile d'Appel	<p>Utiliser cette option avant de créer un diagramme Séquence Pile d'Appel pour limiter la longueur de la pile. Lorsque cette option est sélectionnée, le cadre est marqué et son texte est mis en surbrillance. Les cadres situés au-dessus de celui-ci seront alors exclus de tout diagramme Séquence produit.</p> <p>Cette option n'est pas applicable aux rapports de profil de mémoire.</p>
Supprimer la marque	<p>Supprime la marque d'un cadre qui était précédemment marqué comme « Initial ».</p> <p>Cette option n'est pas applicable aux rapports de profil de mémoire.</p>
Créer Diagramme Pile d'Appel	<p>Génère un diagramme Séquence pour une seule pile dans le graphique. Le cadre sélectionné est représenté comme le cadre terminal de la pile. Le cadre initial de la pile est par défaut le nœud racine si aucun cadre « Initial » n'a été marqué.</p> <p>Cette option n'est pas applicable aux rapports de profil de mémoire.</p>
Créer Diagramme Graphique d'Appel pondéré	<p>Génère un diagramme Séquence qui présente une séquence pour chaque branche de pile visible issue de la trame sélectionnée. En développant et en réduisant les nœuds qui vous intéressent, vous pouvez personnaliser le contenu diagramme Séquence à votre guise.</p> <p>Cette option n'est pas applicable aux rapports de profil de mémoire.</p>
Afficher l'utilisation pondérée la plus lourde	Sélectionnez cette option pour afficher la ligne du graphique avec le poids le plus élevé dans laquelle cette fonction apparaît.
Afficher la prochaine utilisation pondérée	<p>Sélectionnez cette option pour accéder à la ligne suivante du graphique où la fonction apparaît.</p> <p>Vous pouvez utiliser la combinaison de touches de raccourci Ctrl+Flèche vers le bas.</p>

Afficher l'utilisation pondérée précédente	Sélectionnez cette option pour accéder à la ligne précédente du graphique où cette fonction apparaît. Vous pouvez également utiliser la combinaison de touches de raccourci Ctrl+Flèche vers le haut.
Importer le code source	Sélectionnez cette option pour importer le code source sélectionné dans le rapport. Cette option n'est pas applicable aux rapports de profil de mémoire.
Ajustement automatique	Lorsque cette option est activée, elle adapte automatiquement les colonnes à la zone d'affichage disponible.
Enregistrer Rapport dans un fichier	Sélectionnez cette option pour afficher la dialogue « Enregistrer sous », vous permettant de choisir où stocker le rapport.
Enregistrer Rapport sur l'artefact	Note : Avant de sélectionner cette option, allez dans la fenêtre Navigateur et sélectionnez le Paquetage ou l'élément sous lequel créer l'élément Artefact. Vous êtes invité à fournir un nom pour le rapport (et l'élément) ; saisissez-le et cliquez sur le bouton OK . L'élément Artefact est créé dans la fenêtre Navigateur , sous le Paquetage ou l'élément sélectionné. Si vous ajoutez l'artefact à un diagramme sous forme de lien simple, lorsque vous double-cliquez sur l'élément, le rapport est rouvert.

Notes

- Si vous ajoutez le rapport du profileur à un élément d'artefact et joignez également un document lié, le rapport du profileur a la priorité et s'affiche lorsque vous double-cliquez sur l'élément ; vous pouvez afficher le document lié à l'aide de l'option de menu contextuel « Modifier le document lié »

Enregistrer Rapport dans Bibliothèque d'Équipe

Vous pouvez enregistrer n'importe quel rapport actuel comme ressource pour une catégorie, un sujet ou un document dans la Bibliothèque d'Équipe . Le rapport peut ensuite être partagé et examiné à tout moment car il est enregistré avec le modèle. Cela vous aide à :

- Conserver un rapport de profilage pour le comparer aux exécutions futures
- Autoriser d'autres personnes à enquêter sur le profil

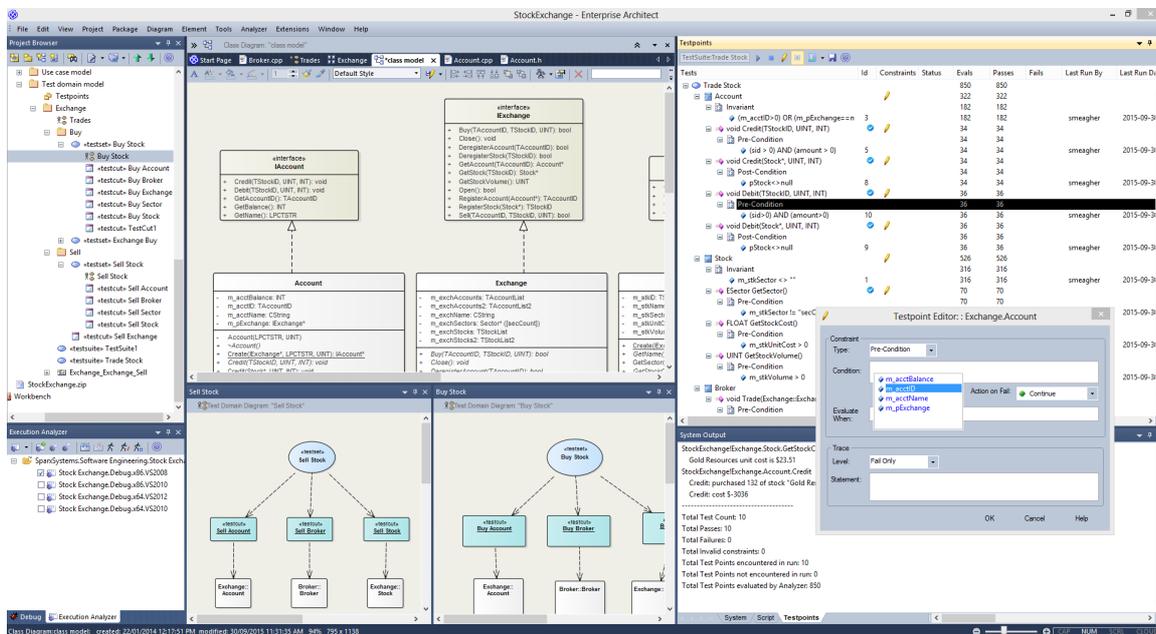
Accéder

Menu Contexte	Cliquez-droit dans la fenêtre Bibliothèque Partager la ressource Rapport Actif Profiler
---------------	--

Testpoints

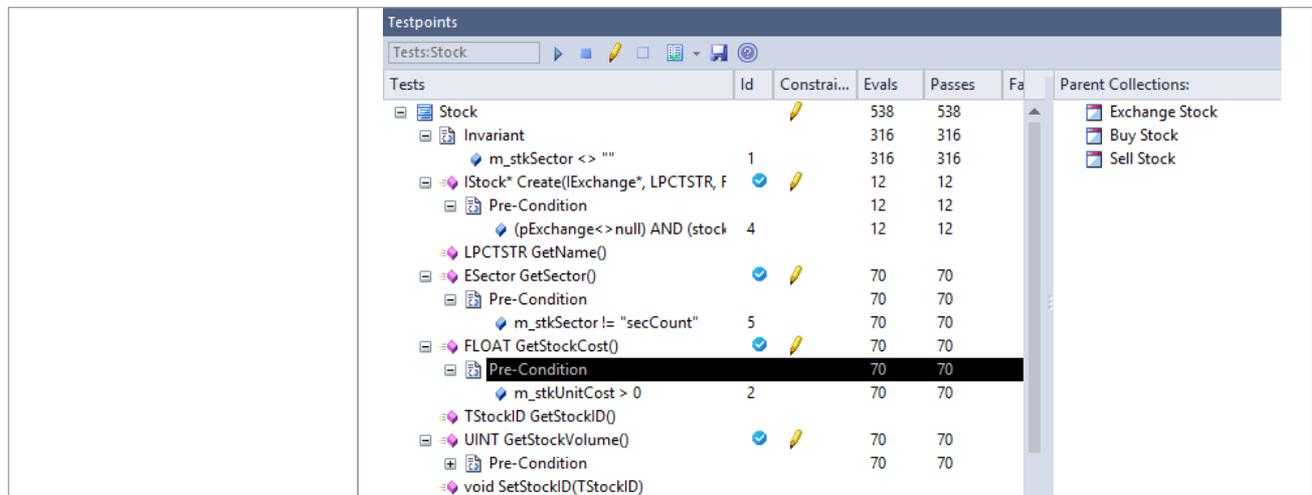
Testpoints présentent un schéma par lequel les contraintes et les règles régissant le comportement des objets peuvent être extraites du modèle et appliquées à une ou plusieurs applications. Les avantages offerts par des schémas tels que celui-ci sont la tolérance aux changements de code : l'ajout et la soustraction de lignes d'une fonction n'ont aucun effet sur les contraintes qui la régissent. Un autre avantage est que les modifications apportées aux règles de comportement ne nécessitent pas de modification correspondante du code source ; *ce qui signifie que rien ne doit être recompilé !*

De plus, la possibilité de vérifier plusieurs applications à l'aide d'un seul domaine de test est une question simple plutôt que contraignante. Le domaine Test est à la fois un modèle logique et relationnel ; les contraintes du modèle de classe peuvent être partitionnées avec des coupes Test . Celles-ci peuvent être agrégées simplement dans Ensembles Test et des suites Test à l'aide de connecteurs. En raison du découplage du domaine Test de la base de code, il suffit de choisir des boutons pour exécuter un programme normalement ou l' exécuter pour un domaine Test spécifique. Ce système offre également des avantages pratiques dans la mesure où aucune instrumentation n'est requise. Les résultats Test sont affichés dans la fenêtre de rapport pendant l' exécution , en temps réel, pendant l'exécution du programme. Ces résultats peuvent être conservés et examinés à tout moment dans la dialogue « Détails Test » ou à l'aide des fonctionnalités de documentation d' Enterprise Architect .



Fonctionnalités

Fonctionnalité	Détails
<p>Composition Testpoint</p>	<p>La composition Testpoint est effectuée à l'aide de la fenêtre Testpoints . La fenêtre Testpoints est contextuelle et affiche le domaine Test de l'élément sélectionné dans la fenêtre Navigateur ou diagramme . La sélection d'une seule classe affiche la structure de la classe. Une icône en forme de crayon s'affiche en regard des classes et des méthodes qui ont des contraintes existantes.</p> <p>Lorsque vous sélectionnez un Test coupe, d'ensemble ou Test suite, la fenêtre Testpoints affiche la structure complète du domaine, y compris toutes les classes qui composent le domaine. Note : vous pouvez parcourir la hiérarchie du domaine à l'aide du volet « Navigation » sur la droite. Testpoints sont composés sous forme d'expressions, à l'aide des noms de variables des membres de la classe. Le raccourci Intelli-sense Ctrl+Espace est disponible dans l'éditeur pour vous aider à les trouver. Les expressions qui donnent la valeur True sont considérées comme une réussite. Le retour de False est considéré comme un échec.</p>



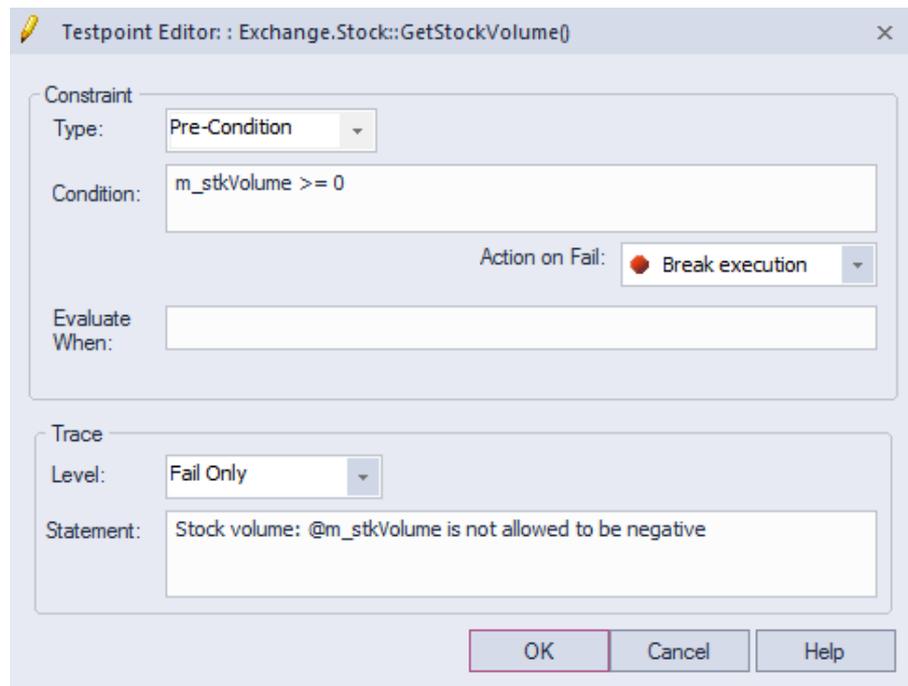
Vous pouvez ajouter ou modifier un invariant existant en double-cliquant sur la classe.

Vous pouvez ajouter ou modifier une pré-condition ou une post-condition existante de la même manière en double-cliquant sur la méthode.

Un double-clic sur un Testpoint affichera automatiquement le code source s'il est disponible.

Il est préférable d'ajouter des conditions de ligne à partir de l'éditeur de code à l'aide de ses menus contextuels.

Cette image représente une condition préalable dans le domaine Test .

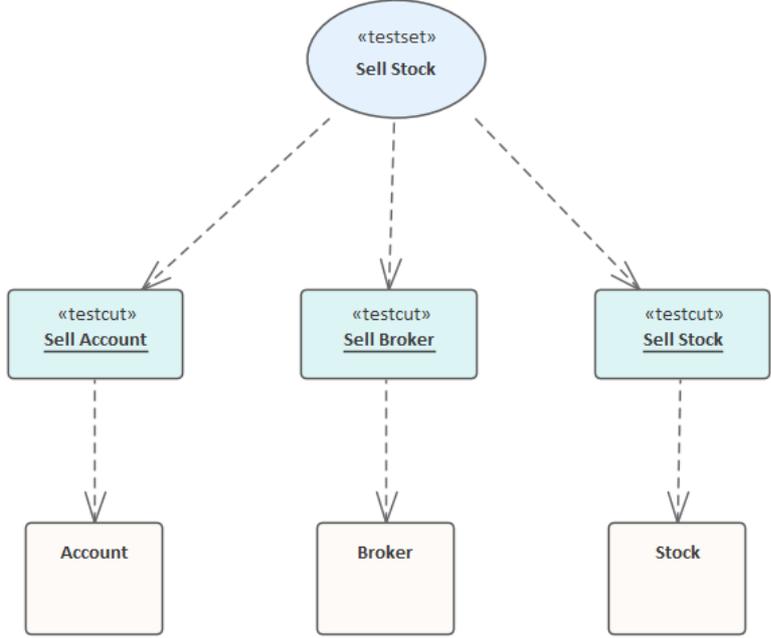
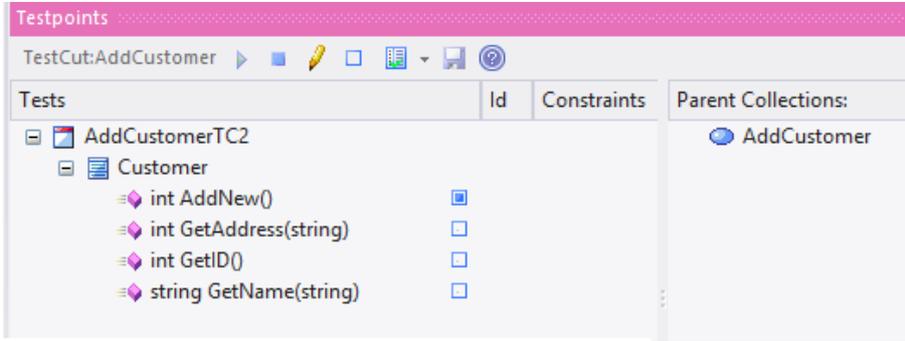


Déclarations de trace de Testpoint

Chaque Testpoint peut avoir sa propre instruction Trace. L'instruction Trace est un message dynamique qui peut référencer des variables dans son objet ou sa portée locale. Elles sont générées lors de l'évaluation d'un test. Elles peuvent être configurées pour être générées à chaque fois qu'une contrainte est évaluée, ou plus généralement lorsqu'un test a échoué. Les instructions Trace peuvent être dirigées vers l'onglet « Testpoints » de la fenêtre Sortie système ou vers un fichier externe. Vous pouvez configurer cela dans n'importe quel script Analyzer.

Composition du domaine

Le diagramme de domaine Test est un support dynamique dans lequel Testpoints

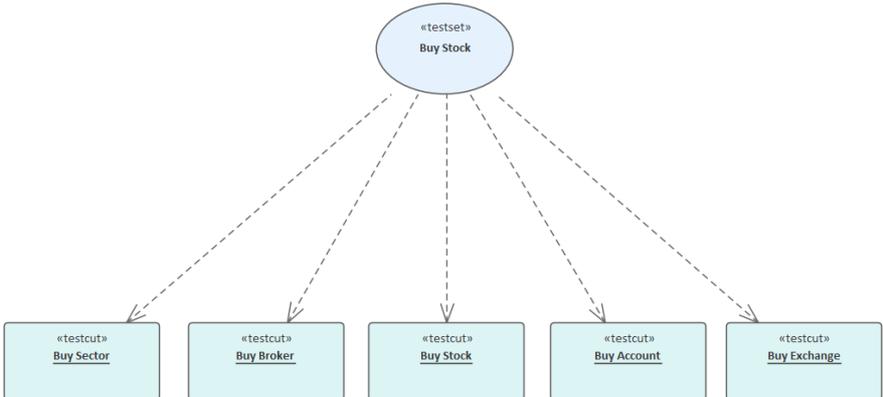
<p>Test</p>	<p>sont assemblés pour tester les cas d'utilisation. Les cas d'utilisation dans un diagramme de domaine Test sont fournis dans trois stéréotypes différents : Test Cut, Ensemble de Test et Suite de Test . La gestion du domaine est aussi simple que modélisation sur n'importe quel diagramme . La boîte à outils et les menus contextuels donnent accès à tous les artefacts du domaine de Test . En bref, Testpoints de plusieurs classes sont agrégés dans Ensembles Test . Ensembles Test sont ensuite liés pour former des suites Test . Les coupes Test et Ensembles Test sont des ressources réutilisables. Lier le même Ensemble de Test à une ou plusieurs suites Test est une question de dessin de connecteurs.</p>  <pre> graph TD TS[«testset» Sell Stock] -.-> TC1[«testcut» Sell Account] TS -.-> TC2[«testcut» Sell Broker] TS -.-> TC3[«testcut» Sell Stock] TC1 -.-> A[Account] TC2 -.-> B[Broker] TC3 -.-> S[Stock] </pre>
<p>Domaine Test et Modèle de classe</p>	<p>Il est rare qu'un cas d'utilisation implique toutes les méthodes d'une seule classe. Il est très probablement réalisé à l'aide d'une variété de méthodes provenant de classes collaboratives. Nous appelons ce sous-ensemble de méthodes une « coupe », et l'artefact de coupe Test est l'outil que nous utilisons pour effectuer ces coupes. La fenêtre Testpoints s'adapte en fonction du contexte, pour être celle d'un domaine Test ou d'un élément de classe. Cette image montre la fenêtre Testpoints lorsqu'une coupe Test a été sélectionnée. Note les cases à cocher, qui ne sont visibles que pour une coupe Test . Elles désignent les méthodes (coupures Test) qui contribuent à un Ensemble de Test . Dans cet exemple, le domaine Test a été généré par l' Analyseur d'Exécution , qui a effectué le travail d'identification des méthodes pour nous.</p> 
<p>Évaluation Testpoint</p>	<p>La fenêtre Testpoints est utilisée pour évaluer les domaines Test . La fenêtre dispose d'une barre d'outils pour démarrer ou se connecter à l'application cible. Le domaine à tester est toujours reflété par l'élément qui a un contexte, donc si vous sélectionnez une classe, la fenêtre affichera uniquement la structure de la classe et</p>

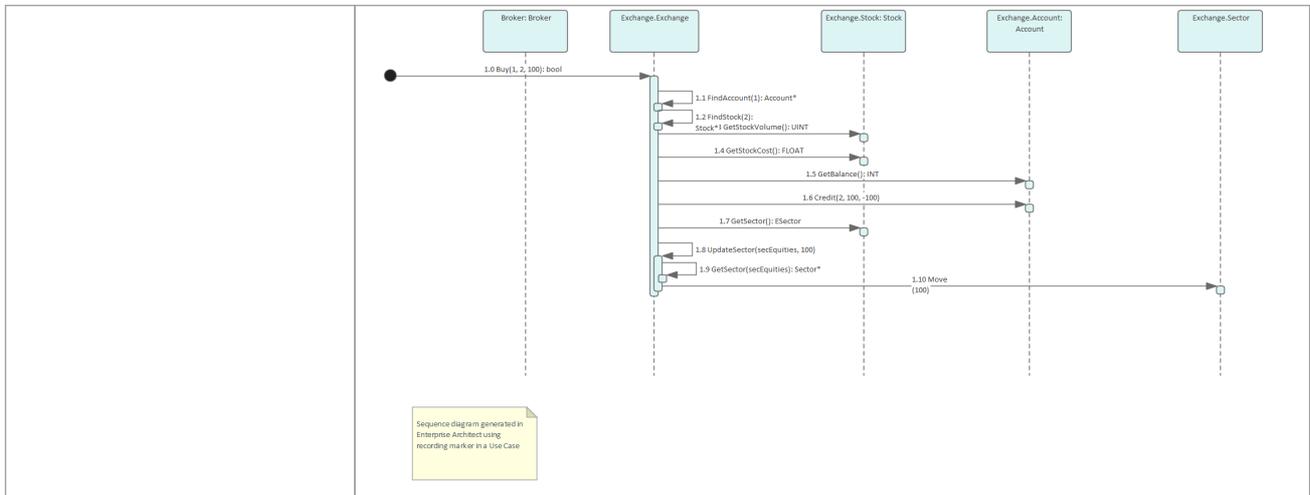
	<p>Testpoints de cette classe. Si vous sélectionnez une Suite de Test , la fenêtre affichera toute la hiérarchie du domaine et tous les Testpoints qu'elle contient. Cliquer sur le bouton Exécuter chargera le domaine Testpoint dans l' Analyseur d'Exécution , qui évaluera, collectera et mettra à jour la fenêtre de rapport au fur et à mesure que les cas d'utilisation réussissent ou échouent à chaque test. Les détails exacts de chaque type de contrainte et le moment et la manière de la capture de cette contrainte sont les suivants :</p> <ul style="list-style-type: none">• Un invariant de classe est évalué par l'analyseur chaque fois qu'une méthode appelée sur un objet de ce type de classe est terminée ; l'invariant sert à tester que l'état d'un objet conforme est à la fois connu et autorisé• Les conditions préalables sont évaluées immédiatement avant l'appel d'une opération• Les post-conditions sont évaluées (en même temps qu'un invariant de classe) lorsque la méthode est terminée• Les conditions de ligne sont évaluées si et quand leur ligne de code spécifique entre dans le champ d'application pendant l'exécution du programme
--	--

Diagramme Domaine de Test

Le diagramme de domaine Test est le support sur lequel vous assemblez et regroupez les cas de test pour un domaine particulier. Un exemple de domaine Test peut être « Client ». L'étendue et la profondeur des domaines que vous assemblez dépendent de vous. Vous pouvez avoir des domaines distincts pour « Ajouter un client » et « Supprimer un client », en fonction de la manière dont vous considérez le mieux pour équilibrer la hiérarchie des domaines. La boîte à outils Diagramme et le menu contextuel fournissent un certain nombre d'artefacts pour vous aider à modéliser le domaine. Étant donné que le support est dynamique, ce qui vous permet de revisiter et de développer les relations entre les domaines Test, le système est un excellent modèle pour fournir des ressources réutilisables à une organisation qui ont une faible charge et s'intègrent à la fois à la vision UML du monde et aux rouages Ingénierie de Logiciel de la vie quotidienne.

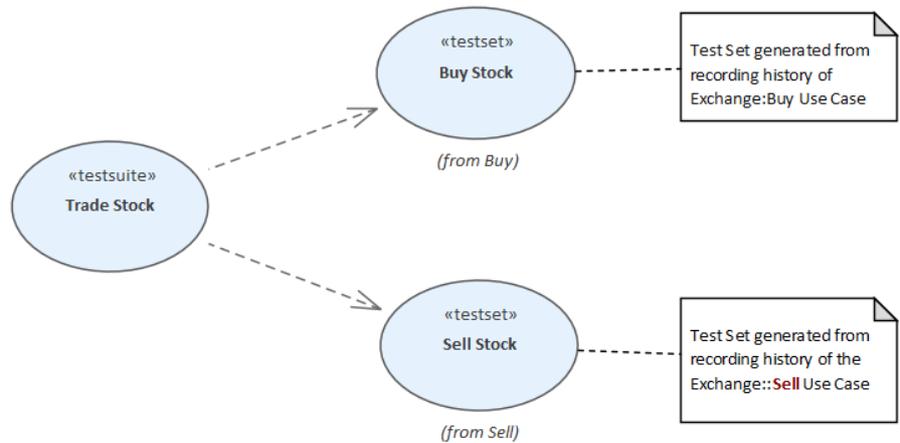
Facilités

Facilité	Détails
Génération de domaine Test	<p>Si vous pensez que le processus de composition d'un domaine Test est complexe, c'est peut-être le cas, mais vous pouvez obtenir de l'aide ! L'Analyseur d'Exécution peut produire un diagramme de domaine Test pour vous. Il ne peut pas écrire les tests à votre place, mais il peut faire une partie du travail. Il peut identifier les classes et sélectionner uniquement les méthodes qui ont participé à un cas d'utilisation. Et ce n'est pas une conjecture. Le domaine Test de l'Analyseur est obtenu à partir d'un programme en cours d'exécution. Cette image montre le domaine de Test généré par l'Analyseur d'Exécution à partir de l'enregistrement d'un programme Modèle d'exemple.</p>  <pre> graph TD A("«testset» Buy Stock") -.-> B["«testcut» Buy Sector"] A -.-> C["«testcut» Buy Broker"] A -.-> D["«testcut» Buy Stock"] A -.-> E["«testcut» Buy Account"] A -.-> F["«testcut» Buy Exchange"] </pre> <p>Et c'est l'enregistrement lui-même (sous forme de diagramme Séquence) à partir duquel le domaine Test a été généré.</p>



Composition du domaine Test

La première tâche sur un diagramme de domaine Test consiste à créer les cas d'utilisation (Ensembles Test). Ceux-ci définissent la responsabilité de ce domaine particulier. La boîte à outils Diagramme et le menu contextuel fournissent des artefacts pour vous aider à y parvenir. Le premier de ces éléments est le Test Cut, qui est utilisé dans l'étape suivante ; identifier les méthodes (à partir du modèle de classe) que vous considérez comme des participants au cas d'utilisation. L'artefact Test Cut est utile car il nous permet de partitionner une classe, en sélectionnant uniquement les méthodes pertinentes. Test Cut peuvent être exécuter individuellement ou liés à un ou plusieurs Test Ensembles . Test Ensembles peuvent à leur tour être liés à une ou plusieurs Test Suites. Dans tous les cas, tout élément de l'arborescence du domaine Test peut être exécuter individuellement ou dans son ensemble.



Coupe Test



Description

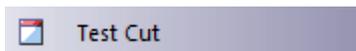
Un élément Test Cut est un élément Object stéréotypé, utilisé en interne dans Enterprise Architect pour définir Ensembles Test à l'aide des facilités de test de code Testpoint .

Une tâche, telle que « Imprimer », peut impliquer des opérations sur différentes classes. Pour créer un test « Imprimer », vous devez inclure uniquement les opérations « Imprimer » de ces classes et exclure toutes les autres opérations.

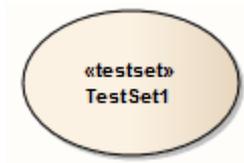
Un Test Cut vous permet de capturer uniquement les opérations qui représentent le comportement (dans ce cas, « Imprimer ») défini pour une seule classe. Vous pouvez ensuite placer le Test Cut de chacune des plusieurs classes dans une seule tâche en tant qu'Ensemble Ensemble de Test .

Lorsque vous faites glisser un élément Test Cut sur un diagramme Test Domain, vous créez une relation de dépendance avec l'élément Class requis. Par conséquent, lorsque vous sélectionnez l'élément Test Cut dans la fenêtre Testpoints , les opérations de la classe sont répertoriées dans la fenêtre, chacune avec une case à cocher. Vous sélectionnez ensuite la case à cocher en regard de chaque opération de classe à inclure dans le Test Cut.

Icône de la boîte à outils



Ensemble de Test



Description

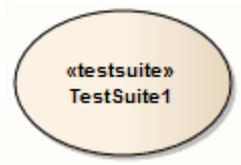
Un élément Ensemble de Test est un élément de cas d'utilisation stéréotypé utilisé pour regrouper un ou plusieurs groupes de méthodes (coupures Test), qui peuvent s'étendre sur plusieurs classes, en une seule tâche. Ensembles Test peuvent également être regroupés en suites Test .

Vous liez les éléments Test Cut à l' Ensemble de Test à l'aide de connecteurs de dépendance.

Icône de la boîte à outils



Suite de Test

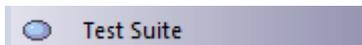


Description

Un élément Suite de Test est un élément Use Case stéréotypé, utilisé pour regrouper un ou plusieurs groupes de tâches (Ensembles Test).

Vous liez les éléments Ensemble de Test à la Suite de Test à l'aide de connecteurs de dépendances.

Icône de la boîte à outils



La Fenêtre Testpoints

La Fenêtre Testpoints est le hub où sont composées les contraintes du Domaine Test . C'est aussi le contrôle qui vous permet de vérifier un Domaine Test particulier sur un programme. Le programme peut être déjà en cours d'exécution ou il peut être lancé à l'aide de la barre d'outils du contrôle. Ici vous pourrez également voir les résultats de vos tests, au fur et à mesure qu'ils se produisent. Ce contrôle est contextuel, réagissant à la sélection d'éléments dans la fenêtre Navigateur ou sur un diagramme . Selon la sélection, les tests peuvent être effectués sur une seule Classe, un Cas d'Utilisation (Ensemble de Test) ou une collection de Cas d'Utilisation (Une Suite de Test).

Accéder

Ruban	Exécuter > Outils > Testeur > Afficher la fenêtre Testpoint
-------	---

Colonnes de la Fenêtre Testpoints

Colonne	Usage
Tests	<p>Affiche le nom de l' object Testpoint sélectionné et la hiérarchie des objets situés en dessous.</p> <p>L' object sélectionné peut être un :</p> <ul style="list-style-type: none"> • Classe • Opération • Coupe Test • Ensemble de Test ou • Suite de Test
Identifiant	<p>Pour une opération, cette colonne affiche une icône de marqueur Testpoint () lorsque l'analyseur a correctement lié cette opération dans l'application cible. Si aucune icône n'apparaît dans cette colonne pendant un exécuter , cela indique que le modèle et la base de code ne sont peut-être pas synchronisés ; peut-être que la signature de la fonction a changé, ou que l'opération est une nouvelle méthode sur laquelle vous travaillez qui existe dans le code source mais pas encore dans votre modèle.</p> <p>Pour un Testpoint , cette colonne affiche un numéro d'identification généré. Ce numéro d'identification est utilisé dans la sortie de trace pour indiquer la contrainte référencée.</p>
Contraintes	<p>Une icône en forme de crayon () dans cette colonne indique qu'une ou plusieurs contraintes sont définies pour cette classe ou opération.</p>
Statut	<p>Lors d'un test exécuter , indique ces statuts possibles :</p> <ul style="list-style-type: none"> • () Échec - La contrainte a été évaluée comme fausse une ou plusieurs fois. • () Déclaration non valide - La contrainte n'a pas pu être analysée en raison d'une syntaxe non valide. • () Variable non trouvée - Aucun nom de variable référencé n'a été trouvé à

	<p>l'emplacement où la contrainte a été évaluée.</p> <p>Aucune icône n'est affichée si une contrainte est réussie.</p>
Évaluations	Lors d'un test exécuter , indique le nombre de fois que l' Analyseur d'Exécution a évalué cette contrainte.
Passes	Lors d'un test exécuter , indique le nombre de fois que le test a réussi.
Échecs	Lors d'un test exécuter , indique le nombre de fois où le test a échoué.
Dernier Exécuter Par	Affiche le nom d'utilisateur de la dernière personne à exécuter ce test. (Les valeurs sont dérivées des définitions de l'auteur du projet dans la dialogue « Personnes » - « Paramètres > Données de référence > Types Modèle > Personnes > Auteurs du projet ».)
Date du dernier Exécuter	Affiche la date et l'heure de la dernière évaluation de ce test.
Résultat du dernier Exécuter	Affiche le résultat du dernier test exécuter .
Volet des collections parentes	<p>Répertorie toutes les collections parentes qui incluent l' object sélectionné dans le cadre de leur conception.</p> <p>Double-cliquez sur cette collection pour en faire l' object sélectionné dans le volet de gauche.</p> <p>Le volet Collections parentes peut être masqué en cliquant sur le bouton Afficher/Masquer le volet Collections parentes dans la barre d'outils de la fenêtre Testpoints .</p>

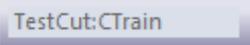
Barre d'Outils Testpoints

La Fenêtre Testpoints fournit des options permettant d'exécuter des tests configurés sur l' object Testpoint actuellement sélectionné, d' exécuter un test en cours, de filtrer les éléments affichés et d' exécuter les résultats d'un test terminé.

Accéder

Ruban	Exécuter > Outils > Testeur > Afficher la fenêtre Testpoint
-------	---

Options de la barre d'outils Testpoints

Bouton de la barre d'outils	Action
	Champ affichant le nom de l' object Testpoint actuellement sélectionné.
	Exécuter le test exécuter .
	Arrêter l' exécuter du test actuellement en cours.
	Basculez entre l'affichage de tous les éléments et l'affichage uniquement des éléments pour lesquels des contraintes sont définies.
	Basculez entre l'affichage de tous les éléments et l'affichage uniquement des opérations qui ont été marquées pour être incluses dans ce Test de coupe ; ce bouton n'est activé que lorsqu'un object Test Cut est sélectionné. Lorsqu'un Test coupe est sélectionné, chacune des opérations de sa classe associée est affichée avec une case à cocher ; vous utilisez cette case à cocher pour marquer les opérations qui s'appliquent à ce Test coupe.
	<p>Cliquez sur la flèche déroulante à côté de cette icône pour afficher le menu « Options Exécuter Test », proposant ces options :</p> <ul style="list-style-type: none"> « Préfixer la sortie de trace avec un appel de fonction » : préfixer toutes les lignes de sortie de trace avec le nom de la fonction en cours d'exécution « Activer Points d'Arrêt standard pendant Tester » : si cette option n'est pas cochée, l' exécuter de test ignore tous les points d'arrêt de l'ensemble de points d'arrêt actuel et toutes les tentatives de définition de points d'arrêt pendant l' exécuter sont ignorées. « Sortie de la trace Vue » - Affiche l'onglet « Testpoints » de la fenêtre Sortie système
	<p>Cliquez sur cette icône après avoir terminé un exécuter pour enregistrer les résultats dans l'élément Test sur l' object actuel. Les tests enregistrés peuvent être visualisés à l'aide de l'espace de travail Tester .</p> <p>Une prompt s'affiche pour sélectionner la classe Test : Unité, Intégration, Système, Inspection, Acceptation ou Scénario. Sélectionnez la classe Test appropriée et</p>

	cliquez sur le bouton OK .
	Afficher la rubrique d'aide sur la gestion Testpoint .
	Afficher ou masquer le volet Collections parentes.

Éditeur Testpoints

L'Éditeur Testpoints permet de composer des contraintes pour les Classes et les Opérations. Les types de contraintes autorisés dépendent de l'objet sélectionné. Pour les Classes, le type sera toujours Invariant. Pour les Opérations, le type peut être soit Pre-Condition, Post-Condition ou Line-Condition.

Les invariants sont évalués par l'analyseur lorsqu'une méthode appelée sur un objet du type de classe sélectionné se termine. Les pré-conditions sont évaluées au début de chaque appel à l'opération spécifiée. Les post-conditions sont évaluées à la fin de chaque appel à l'opération spécifiée. Les conditions de ligne sont évaluées chaque fois que la ligne de code spécifiée est exécutée.

Accéder

Ruban	<ol style="list-style-type: none"> 1. Exécuter > Outils > Testeur > Afficher la fenêtre Testpoint . 2. Dans la fenêtre Testpoints , double-cliquez sur une Classe ou une Opération pour afficher la dialogue « Éditeur Testpoints ».
-------	---

Champs du groupe de contraintes

Champ	Usage
Type	Le type de contrainte pour la classe ou l'opération sélectionnée : <ul style="list-style-type: none"> • Invariant - Évalué après la fin de toute méthode appelée sur la classe spécifiée • Pré-condition - Évaluée au début de chaque appel à une opération spécifique • Post-Condition - Évalué après la fin de chaque appel à une opération spécifique

	<ul style="list-style-type: none"> Condition de ligne - Évaluée lors de l'exécution d'une ligne de code spécifique dans une opération
Compenser	<p>Pour les conditions de ligne uniquement, le numéro de ligne dans l'opération spécifiée sur laquelle évaluer la contrainte.</p> <p>Une valeur de décalage est automatiquement définie si le Testpoint a été créé à l'aide du menu contextuel de Éditeur de Code .</p>
Condition	La contrainte à évaluer lorsque ce Testpoint est déclenché. Un statut de réussite ou d'échec sera enregistré selon que cette condition de contrainte est évaluée comme vraie ou fausse.
Action en cas d'échec	<p>Cliquez sur la flèche déroulante et sélectionnez parmi les trois options :</p> <ul style="list-style-type: none"> « Continuer » : ignorer l'échec de cette contrainte et continuer l'exécution « Interrompre l'exécution » - interrompre l'exécution et afficher la trace de la pile « Désactiver en cas d'échec » : ne pas exécuter à nouveau la contrainte après un échec
Évaluer quand	(Facultatif) Une contrainte supplémentaire qui doit être respectée avant que la condition Testpoint principal ne soit évaluée, offrant un meilleur contrôle sur la couverture du test.

Champs du groupe de trace

Option	Action
Niveau	<p>Spécifie quand l'instruction de trace (si elle est définie) sera générée. Les options disponibles sont :</p> <ul style="list-style-type: none"> « Échec uniquement » : instruction de trace de sortie uniquement lorsque cette condition Testpoint échoue « Toujours » : Instruction de trace de sortie à chaque fois que ce Testpoint est évalué
Déclaration	<p>(Facultatif) Un message à afficher lorsque ce Testpoint est évalué.</p> <p>Les variables actuellement dans la portée peuvent être incluses dans la sortie d'une instruction de trace en préfixant le nom de la variable avec un jeton \$ pour les variables string , ou un jeton @ pour les types primitifs tels que ' int ' ou 'long ' .</p> <p>La sortie d'une instruction Trace peut être dirigée soit vers l'onglet « Testpoints » de la fenêtre de sortie système, soit vers un fichier externe, tel que configuré par le script Analyzer pour le Paquetage parent.</p>

Contraintes Testpoint

Une contrainte est généralement composée de variables locales et membres dans des expressions, séparées par des opérateurs pour définir un ou plusieurs critères spécifiques qui doivent être respectés. Une contrainte doit être évaluée comme vraie pour être considérée comme réussie. Si une contrainte est évaluée comme fausse, elle est considérée comme échouée.

Toutes les variables référencées dans la contrainte doivent être dans la portée à la position où le Testpoint ou Point d'Arrêt est évalué.

Opérateurs généraux/ Arithmétique

Opérateur	Description
+	Ajouter Exemple : $a + b > 0$
-	Soustraire Exemple : $a - b > 0$
/	Diviser Exemple : $a / b == 2$
*	Multiplier Exemple : $a * b == c$
%	Module Exemple : $a \% 2 == 1$
()	Parenthèses - Utilisées pour définir la priorité dans les expressions complexes. Exemple : $((a / b) * c) <= 100$
[]	Crochets - Utilisés pour accéder aux tableaux. Exemple : <code>Names[0].Surname == « Smith »</code>
.	Opérateur point - Utilisé pour accéder aux variables membres d'une classe. Exemple : <code>Station.Name == « Flinders »</code>
->	Notation alternative pour l'opérateur Point. Exemple : <code>Station->Nom == « Flinders »</code>

Opérateurs de comparaison

Opérateur	Description

=	Égal à Exemple : a = b
==	Égal à Exemple : a == b
!=	Pas égal à Exemple : a != b
<>	Pas égal à Exemple : a <> b
>	Plus grand que Exemple : a > b
>=	Supérieur ou égal à Exemple : a >= b
<	Moins que Exemple : a < b
<=	Inférieur ou égal à Exemple : a <= b

Opérateurs logiques

Opérateur	Description
AND	AND logique Exemple : (a >= 1) AND (a <= 10)
OR	OR logique Exemple : (a == 1) OR (b == 1)

Opérateurs au niveau du bit

Opérateur	Description
&	AND au niveau du bit Exemple : (1 & 1) = 1 (1 et 0) = 0
	OR au niveau du bit

	Exemple : $(1 1) = 1$ $(1 0) = 1$
\wedge	XOR au niveau du bit (OR exclusif) Exemple : $(1 \wedge 1) = 0$ $(1 \wedge 0) = 1$

Exemples supplémentaires

Exemple	Description
$((m_nValeur \& 0xFFFF0000) == 0)$	Utilisez un opérateur AND au niveau du bit (&) avec une valeur hexadécimale comme opérande de droite pour tester qu'aucun bit n'est défini dans les octets d'ordre supérieur de la variable.
$((m_nValue \& 0x0000FFFF) == 0)$	Utilisez un opérateur AND au niveau du bit (&) avec une valeur hexadécimale comme opérande de droite pour tester qu'aucun bit n'est défini dans les octets de poids faible de la variable.
$m_value[0][1] = 2$	Accéder à un tableau multidimensionnel
$a \text{ AND } (b \text{ OR } c)$	Combinaison des opérateurs AND et OR , en utilisant des parenthèses pour garantir la priorité. Dans cet exemple, la variable « a » doit être vraie et « b » ou « c » doit être vraie.

Notes

- Les comparaisons String sont sensibles à la casse

Test Unitaire

Enterprise Architect supporte l'intégration avec les outils de tests unitaires afin de faciliter le développement de logiciels de bonne qualité.

Dans l'ordre :

- Vous téléchargez et installez les applications NUnit et JUnit (NUnit - <http://www.nunit.org/> JUnit - <http://www.junit.org/>); Enterprise Architect n'inclut pas ces applications dans le programme d'installation
- Enterprise Architect vous aide à créer des stubs de classe de test avec les transformations NUnit et JUnit
- Vous définissez votre code de test dans les stubs de classe
- Vous configurez et exécutez un script de test sur n'importe quel Paquetage
- Tous les résultats des tests sont automatiquement enregistrés dans Enterprise Architect

Configurer Tests Unitaires

Cette rubrique explique les actions à effectuer lors de la configuration Test Unitaire , après avoir téléchargé et installé les applications JUnit et/ou NUnit.

Actes

Action	Détails
Créer des stubs Test unitaires	<p>En utilisant les transformations JUnit ou NUnit et la génération de code, vous pouvez créer des stubs de méthodes de test pour toutes les méthodes publiques de chacune de vos classes.</p> <pre>(TestFixture) classe publique CalculatorTest { (Test) public void testAdd(){ } (Test) public void testDivide(){ } (Test) public void testMultiply(){ } (Test) public void testSubtract(){ } }</pre>
Définir les cas Test	<p>Écrivez votre test unitaire dans les stubs de code générés (soit dans Enterprise Architect , soit dans votre IDE préféré).</p> <p>Il s'agit d'un exemple NUnit en C# , bien qu'il puisse également s'agir de n'importe quel autre langage .NET , ou de Java et JUnit.</p> <pre>(TestFixture) classe publique CalculatorTest { (Test) public void testAdd(){ Assert.AreEqual(1+1,2); } (Test) public void testDivide(){ Assert.AreEqual(2/2,1); }</pre>

	<pre>} (Test) public void testMultiply(){ Assert.AreEqual(1*1,1); } (Test) public void testSubtract(){ Assert.AreEqual(1-1,1); } }</pre> <p>Sinon, si vous n'avez pas effectué de transformation xUnit, vous pouvez procéder à une rétro-ingénierie du code dans Enterprise Architect afin que le système puisse enregistrer tous les résultats des tests par rapport à cette classe.</p>
Compilez votre code	Vérifiez que le code source testé se compile sans erreurs, afin que les scripts de test puissent être exécuter dessus.
Configurer les Scripts de Test	Configurez les scripts Test par rapport au Paquetage requis, puis exécuter les tests.

Exécuter Tests Unitaires

Lors de l'exécution d'un script de test, vous générez des résultats de test qui sont stockés sous forme de cas Test par rapport aux classes testées.

Accéder

Ruban	Exécuter > Exécuter > Démarrer > Test
Analyseur d'Exécution Window	ToolBar > Exécuter Script de Test Menu Contexte > Test

Tâches

Tâche	Détails
Exécuter des tests	<p>Sélectionnez le Paquetage approprié dans la fenêtre Navigateur .</p> <p>Sélectionnez l'option ' Exécuter Script de Test ' pour exécuter le script de test que vous avez précédemment configuré pour ce Paquetage , dans l' Analyseur d'Exécution .</p>
Résultats Vue	<p>Les résultats des tests xUnit sont affichés dans la fenêtre Sortie système, identifiant les tests exécuter et ceux qui ont échoué.</p> <p>Les résultats montrent également quelle méthode a échoué, ainsi que le fichier et le numéro de ligne où l'échec s'est produit.</p> <p>Double-cliquez sur un message d'erreur ; Enterprise Architect ouvre l'éditeur de cette ligne de code, vous permettant de trouver et de corriger rapidement l'erreur.</p> <p>Enterprise Architect enregistre également l'état exécuter de chaque test par rapport à la classe testée ; ceux-ci sont stockés dans l'élément Cas Test .</p> <p>Un diagramme contenant la classe peut être configuré pour afficher ces cas Test , en exposant le compartiment des scripts de test sur les éléments diagramme .</p>

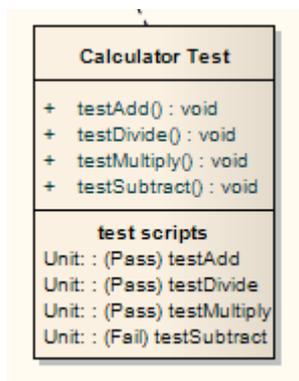
Enregistrer Résultats des Tests

Enterprise Architect est capable d'enregistrer automatiquement tous les résultats des tests via un script de test dans Enterprise Architect .

Processus

Pour utiliser cette fonctionnalité , vous devez procéder à la rétro-ingénierie de la classe de test dans le Paquetage contenant votre script de test.

Une fois que votre modèle contient votre classe de test, lors de la prochaine exécution du script de test, Enterprise Architect ajoute des cas Test à la classe pour chaque méthode de test trouvée ; lors de cette exécution de test et de toutes les suivantes, tous les cas Test sont mis à jour avec l'heure exécution actuelle et s'ils ont réussi ou échoué, comme indiqué :



Calculator Test	
+ testAdd() : void	
+ testDivide() : void	
+ testMultiply() : void	
+ testSubtract() : void	
test scripts	
Unit : (Pass) testAdd	
Unit : (Pass) testDivide	
Unit : (Pass) testMultiply	
Unit : (Fail) testSubtract	

La description de l'erreur pour chaque test ayant échoué est ajoutée à tous les résultats existants pour ce cas Test , ainsi que la date et l'heure actuelles.

Au fil du temps, cela fournit un log de toutes les exécutions de test où chaque cas Test a échoué, qui peut ensuite être inclus dans la documentation générée, ressemblant à ceci :

Échec le 05/07/2006 à 13:02:08

attendu : <0>

mais c'était : <1>

Échec le 28 juin 2006 à 08:45:36

attendu : <0>

mais c'était : <2>

Échantillons

Enterprise Architect vous permet d'importer facilement des exemples de modèles complets (Paquetages), y compris toutes les informations de modèle nécessaires, le code et les scripts de construction. Ces exemples de Motifs facilitent l'exploration et l'essai de l' Analyseur d'Exécution Visuelle . Vous pouvez générer un exemple de modèle pour :

- Java
- Microsoft.NET
- Microsoft C++
- PHP Apache

Accéder

Ruban	Développer > Code source > Créer à partir de Motif > Exemples VEA
-------	---

Exemples d'affichage

Champ	Action
Technologie	Sélectionnez la technologie appropriée.
Nom	Affiche les échantillons disponibles pour la technologie sélectionnée ; sélectionnez l'échantillon requis à importer.
champ de description	Affiche une description de l'échantillon sélectionné.
Dossier de destination	Recherchez et sélectionnez le répertoire dans lequel charger le code source de l'exemple.
Utiliser le chemin local	Activez la sélection d'un chemin local existant pour placer le code source sous ; modifiez le champ « Dossier de destination » en une sélection déroulante.
Commande Compilateur	Affiche le chemin de commande du compilateur par défaut pour la technologie sélectionnée ; vous devez : <ul style="list-style-type: none"> • Confirmez que le compilateur peut être trouvé à ce chemin, ou • Modifier le chemin d'accès à l'emplacement du compilateur
Modifier les chemins locaux	De nombreux exemples VEA spécifient leur compilateur à l'aide d'un chemin local. La première fois que vous utilisez un échantillon, vous devez cliquer sur ce bouton pour afficher la dialogue « Chemins locaux », dans laquelle vous vérifiez et, si nécessaire, corrigez le chemin local pointant vers l'emplacement correct du compilateur.

Notes

- Si nécessaire, vous pouvez définir des exemples personnalisés en ajoutant des fichiers au répertoire AppSamples dans lequel Enterprise Architect est installé ; les répertoires de niveau supérieur sont répertoriés comme Technologies et peuvent contenir un fichier d'icône pour personnaliser l'icône affichée pour la technologie. Les répertoires situés en dessous sont définis comme des groupes dans la liste Motifs ; les Motifs sont définis par la présence de quatre fichiers avec un nom correspondant : un fichier zip (.zip), un fichier XMI (.xml), un fichier de configuration (.cfg) et une icône facultative (.ico)
- Le fichier de configuration supporte ces champs :
 - [fournisseur], [langue], [plateforme], [url], [description], [version] - tous affichés dans la « description » champ
 - [xmrootpaths] - le chemin racine du code source dans le XMI exporté ; il est remplacé par le dossier de destination sélectionné lorsque l'utilisateur applique le motif d'application

