CANTATA

Les nouveautés de Cantata 9.0?

Cantata 9.0, disponible à partir d'Octobre 2019, est une release majeure proposant des fonctions avancées telles que le support du process « Test Driven Development » (TDD) ainsi que l'extension d'AutoTest pour le C++.

Ce document détaille les principales évolutions de Cantata version 9.0.

Introduction

Cantata 9.0, disponible à partir d'Octobre 2019, est une release majeure proposant de nouvelles fonctions avancées. Cette version de l'outillage introduit le support du process « Test Driven Development » (TDD) et améliore la fonction AutoTest en permettant désormais la génération automatique de tests pour le code C++.

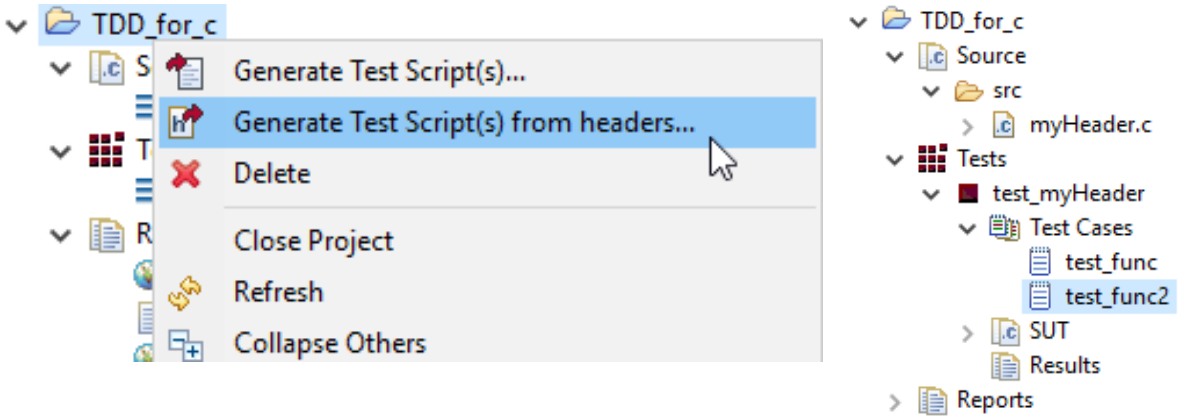
Cantata 9.0 contient également d'autre autres améliorations et corrections de bugs. L'ensemble des modifications apportées au produit sont détaillés dans le document « Release Notes » (qui contient l'historique des changements de Cantata depuis la version 4.1). Les modifications principales de la version 9.0 sont mises en avant et détaillées dans les chapitres ci-dessous.

Support du processus de "Test Driven Development"

Le support du processus de Test Driven Development (TDD) est désormais implémenté dans Cantata. TDD est une méthode de développement pour laquelle les tests unitaires sont créés en première instance et en regard desquels le code source est implémenté. Cette méthode agile peut être utilisée pour écrire un code plus "propre" dans la mesure où l'accent est mis sur la conception des tests à partir des exigences fonctionnelles, au lieu de démarrer directement par l'écriture du code source.

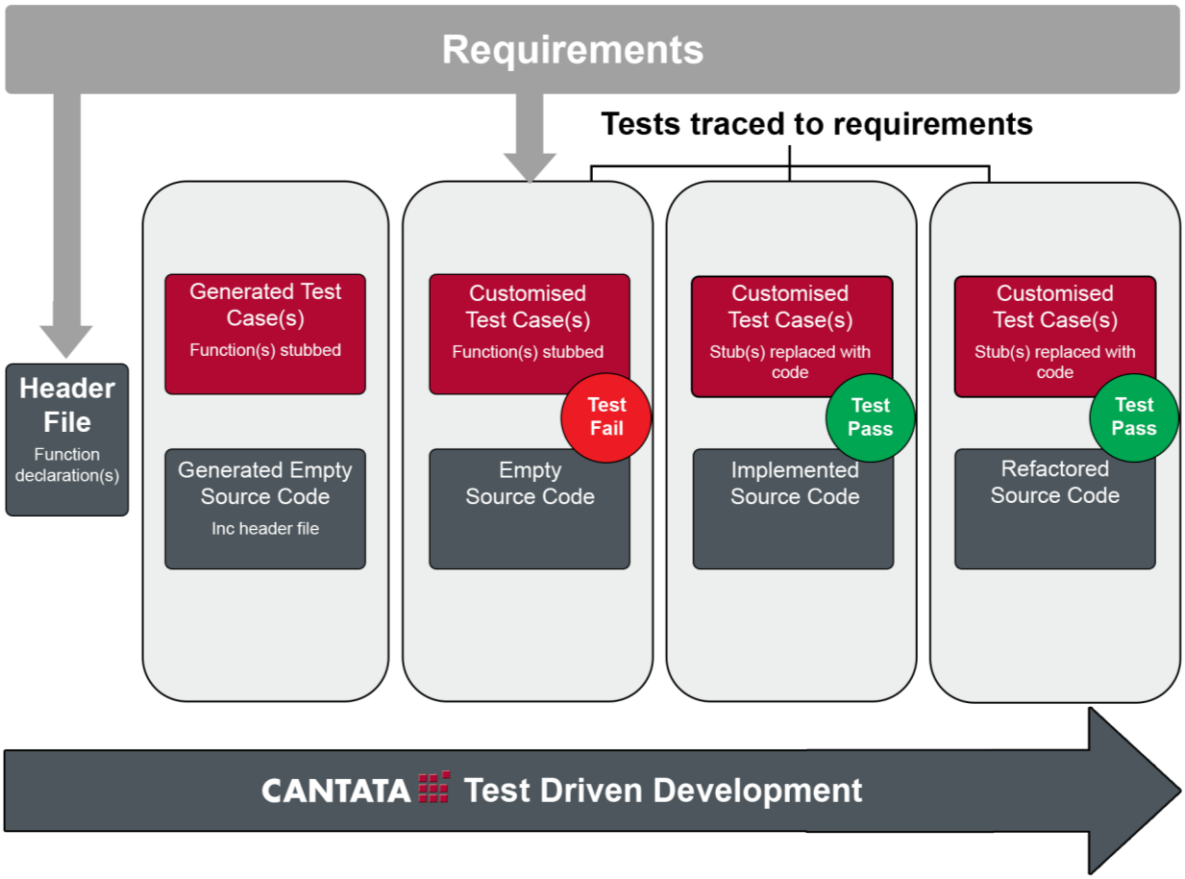
Cantata 9.0 offre de nouvelles fonctions pour le support du processus TDD afin de simplifier la création des tests pour du code source dont l'implémentation n'a pas démarrée.

La fonction TDD peut être utilisée pour générer des cas de tests à partir du moment où les prototypes des fonctions sont disponibles dans des fichiers de déclarations. Un projet standard Cantata est créé, afin de proposer l'environnement de test ainsi que l'ensemble des rapports utilisés comme preuves de certification, et tels qu'attendus dans Cantata. Utiliser Cantata pour le processus TDD améliore l'approche de test unitaire traditionnelle de type boite noire, tout en garantissant l'accès aux fonctions de tests boite blanche car Cantata peut directement accéder à des éléments enfouis du code, tels que les données/fonctions privées et statiques.



La génération des scripts TDD depuis la vue Cantata Test Explorer

Les Scripts de tests sont automatiquement générés sur la base d'un cas de test pour chaque prototype de fonction déclaré dans les fichiers header. Ces cas de tests constituent la première base de vérification, et permettent d'éviter d'avoir à saisir manuellement les informations relatives aux prototypes de fonctions. Chaque donnée globale déclarée dans les fichiers header est détectée automatiquement et vérifiée. La fonction TDD de Cantata génère des fichiers sources vides, de telle sorte que lorsque les corps de fonctions sont implémentés dans un deuxième temps, les tests peuvent être rapidement rejoués afin de vérifier le nouveau code implémenté.



La fonction Cantata Trace peut être utilisée conjointement à la fonction Cantata TDD afin d'améliorer la conception des cas des tests à partir des exigences fonctionnelles. Les exigences importées dans l'outil peuvent être associées aux cas de tests pendant leur conception. Les liens entre les cas de tests, le code et les exigences fonctionnelles sont une garantie de clarté et simplifient ultérieurement la refactorisation du code.

Des bouchons de simulation sont générés pour toutes les fonctions dont le corps n'a pas encore été codé. Cela permet de rendre les cas de tests exécutables même si l'intégralité du code source n'est pas encore disponible, ni même écrite. Ces tests préliminaires vont probablement échouer jusqu'à l'implémentation du code approprié. Une fois que le code est écrit, ces bouchons peuvent être retirés ou convertis en wrappers Cantata (afin d'intercepter et de vérifier les appels aux fonctions / objets) de manière automatique grâce à la fonction d'Analyse des Changements du Code offerte par Cantata. Lorsque les tests passent, la refactorisation du code source peut avoir lieu, et les tests rejoués automatiquement afin de garantir la non-régression.

AutoTest pour le code C++

La fonction Cantata AutoTest a été étendue afin de supporter le code C++. AutoTest génère automatiquement des cas des test unitaires permettant de couvrir du code source C++, et ce pour de nouveaux ou d'anciens projets de développement logiciel, afin de simplement :

- Configurer la génération automatique des tests
  - Choisissez la manière dont vos branches d'exécution seront exercées, grâce à la sélection de différentes règles de couverture de code
  - Configurez le type de test, à travers les différentes options de génération pour les bouchons/wrappers, vérifications des données, etc.
- Générer des scripts de tests permettant d'obtenir 100% de couverture de code
- Suppléer aux manques de couverture de code, une fois que les cas de tests liés aux exigences fonctionnelles ont été exécutés, en activant le rajout automatique de cas de tests AutoTest pour les fonctions et méthodes qui n'ont pas encore été couvertes.
- Créer une base de tests unitaires à des fins de non-régression
- Migrer depuis d'autres solutions de test vers Cantata

AutoTest analyse le code source C++ afin de déterminer l'ensemble des chemins d'exécution sur la base de métriques de couverture de code prédéfinies (telles que 100% des points d'entrée des fonctions, 100% des instructions, 100% des décisions, ou encore 100% de MC/DC). Un algorithme crée les vecteurs de tests afin d'exécuter les différentes branches logiques, en utilisant les capacités avancées de test boite blanche Cantata qui permettent de contrôler les données, paramètres et les interfaces des appels.

Cantata Test Results Summary

Project: *AutoTest Multi File Demo* **Overall Result: Pass**

Summary Information

Hostname	Matt	**Cantata version**	v9.0
Summary generated	Jan 17, 2019, 11:30 AM	**Time elapsed during test run**	16 seconds

Build Summary

Total tests	4
Compile attempted (files)	8
Link attempted (tests)	4
Execute attempted (tests)	4

Results Summary

PASSED	4
FAILED	0
Total checks	233
Total checks failed	0
Total script errors/call failures	0

Coverage Summary

Entry point (E)	100%
Statement (S)	100%
Decision (D)	100%
Call-return (C)	100%
MC/DC - masking (M)	100%
MC/DC - unique cause (U)	100%

Les vecteurs de tests contrôlent l'exécution du code, et vérifient les paramètres échangés entre les méthodes, les valeurs des données globales, l'ordonnancement des appels ainsi que les valeurs de retour des fonctions. Le cas de tests générés par AutoTest peuvent être édités par l'utilisateur comme tous les autres tests Cantata. Chaque cas de test comporte lors de sa création une description du chemin logique exercé dans le code, ce qui simplifie les efforts de maintenance ainsi que la traçabilité vers les exigences fonctionnelles.

AutoTest dans la version 9.0 de Cantata supporte les éléments suivants du langage C++ 03 (et versions antérieures) ;

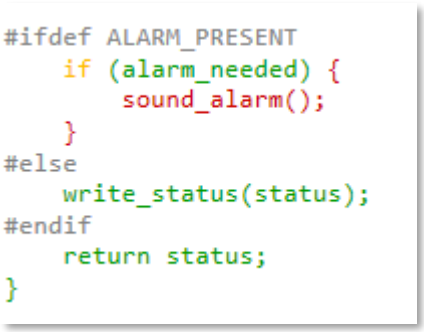
- Classes de base C++ abstraites et concrètes
- Mécanismes d'héritage et de surcharge
- Espaces de nommage
- Gestion des exceptions
- Templates lorsque instanciés dans le code sous test
- Bases mixtes de code C et C++

Cantata AutoTest peut être utilisé sous certaines conditions d'usage avec du code C++ contenant des éléments syntactiques/sémantiques/versions autres que ceux contenus dans la liste ci-dessus. Les tests pour les éléments supportés du langage C++ seront auto-générés, et des tests supplémentaires peuvent être créés afin d'obtenir le niveau de couverture de code désiré.

Amélioration de la Couverture de Code

La Couverture de Code pour les Variantes de Configuration Logicielle

La fonction de Couverture de Code pour les Variantes de Configuration Logicielle C/C++ a été implémentée afin de supporter l'agrégation des données de couverture collectées à travers plusieurs processus de build. Cantata analyse à cette fin les macros utilisées par le préprocesseur (#define) permettant d'identifier les variantes du processus de construction du code. Le visualisateur graphique de Cantata pour la Couverture de Code peut désormais agréger les données pour le même code source à travers plusieurs exécutions correspondant à des variantes de build. Un rapport textuel peut également être généré pour récapituler les directives du préprocesseur ayant été utilisées (ou ignorées). Ce rapport est utilisable dans un processus de certification ayant pour objet des variantes du code source.



```
#ifdef ALARM_PRESENT
    if (alarm_needed) {
        sound_alarm();
    }
#else
    write_status(status);
#endif
    return status;
}
```

Ci-dessus : Visualisateur Graphique de Couverture illustrant du code exécuté pour plusieurs variantes de la macro du préprocesseur intitulée 'Alarm Present'

A droite: Rapport textuel certifiable

```
================================================================================
= Cantata Test Harness v9.0                                                    =
= (c) 2019 QA Systems GmbH                                                     =
=-------------------------------------------------------------------------------=
= Cantata Build Variant Report                                                 =
================================================================================

Input Preprocessor Log Files:
/home/liam/test_cpp_preproc/tmptest/expected_test_file_1.cpl

------------------------- File: 1. test_file_1.c
Preprocessor Directives:
test_file_1.c (7) else
test_file_1.c (11) ifdef DEF_2 >> NOT INCLUDED
test_file_1.c (13) else
test_file_1.c (17) ifndef DEF_4
test_file_1.c (19) else >> NOT INCLUDED
...
------------------------- File: 2. test_file_2.c
Preprocessor Directives:
test_file_2.c (5) if DEF_0 >> NOT INCLUDED

================================================================================
= File                                  Included  Not Included    RESULT =
=-------------------------------------------------------------------------------=
= 1. test_file_1.c                          12            26   >> FAIL =
= 2. test_file_2.c                           0             1   >> FAIL =
=-------------------------------------------------------------------------------=
= TOTALS                                    12            27   >> FAIL =
================================================================================
```

Les rapports d'Analyse Différée de Couverture sont certifiables

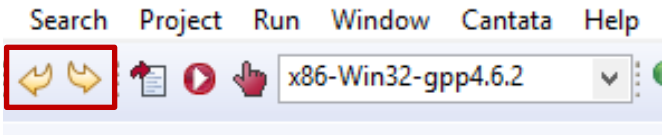
Le mécanisme d'analyse différée de Cantata transfère les données de couverture collectées depuis la cible d'exécution vers la machine hôte afin de procéder aux vérifications de couverture de code et de générer les rapports correspondants. Ce mécanisme est très utile pour les cibles disposant de peu de mémoire. L'utilitaire Cantata intitulé CPPGETCOV utilisé à cette fin fait désormais partie des éléments certifiés par l'organisme indépendant SGS-TÜV SAAR GmbH. Cela implique que les rapports au format. ctr obtenus avec cet utilitaire peuvent être utilisés pour obtenir les crédits de certification adéquats.

Les Résultats de Couverture au format HTML

Les résultats de Couverture de Code peuvent être désormais générés au format HTML à partir des données disponibles dans le visualisateur graphique de Cantata pour la Couverture de Code.

Annuler & Rétablir

Tous les changements apportés aux scripts de tests au travers des différentes vues Cantata sous Eclipse (c.a.d. Test Explorer, Test Case Editor, ou encore Test Properties) sont désormais automatiquement sauvegardés. La fonctionnalité Annuler/Rétablir (Undo/Redo) a donc pu être ajoutée aux vues Cantata pour corriger et refaire les actions de l'utilisateur, qui peut décider d'employer des boutons de l'interface graphique, ainsi que des raccourcis clavier : ctrl+Z (undo) ou bien ctrl+Y (redo).



Mise à jour des Plateformes Supportées

Ainsi que pour toute nouvelle version de Cantata, les plateformes supportées sont mises à jour et étendues.

Cantata est intégré avec les principaux environnements de développement croisé utilisant une base Eclipse (Built-on-Eclipse®), ainsi qu'avec les chaines d'outillage disponibles en tant que plug-ins Eclipse-Ready®. Cantata 9.0 est intégré avec la release Eclipse 2018-09 release (Eclipse 4.9), et est également installable en tant que plug-in Eclipse-Ready® depuis la release Juno (4.2) jusqu'à la release 2018-09 (4.9), ce qui permet à nos clients d'accéder à un riche écosystème d'outils intégrés (par ex. chaines pour ALM, CI et SCM).

Le support des compilateurs GNU GCC et g++ ont été mis à jour jusqu'à la version 8.2 sous Windows, et 8.3 sous Linux.

Le support des différentes constructions du langage C++ 11 & 14 a été également étendu dans Cantata 9.0. Pour obtenir une liste des différentes constructions C++ 11 et C++ 14 (avec des hyperliens vers les descriptions détaillées fournies par open-std.org), veuillez contacter sales@qa-systems.com.