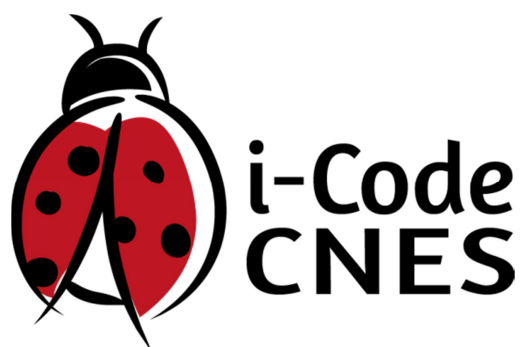


Manuel Utilisateur

i-Code CNES

DCT/AQ /SO - 2014.0020026

Version 1.2



**CENTRE NATIONAL
D'ÉTUDES SPATIALES**

Siège
2, place Maurice Quentin
75039 Paris cedex 01
tél. : 33 (0)1 44 76 75 00

Direction des lanceurs
52, rue Jacques Hillairet
75612 Paris cedex
tél. : 33 (0)1 80 97 71 11

Centre spatial de Toulouse
18, avenue Edouard Belin
31401 Toulouse cedex 9
tél. : 33 (0)5 61 27 31 31

Centre spatial guyanais
BP 726
97387 Kourou cedex
tél. : 33 (0)5 94 33 51 11



TABLE DES MATIERES

1. INTRODUCTION.....	5
2. DOCUMENTS DE REFERENCE	5
3. CONFIGURATION DE L'OUTIL.....	6
3.1. ACTIVATION / DESACTIVATION D'UN LANGAGE.....	6
3.2. ACTIVATION / DESACTIVATION DE LA VERIFICATION DU RESPECT DES REGLES DE CODAGE ET DU CALCUL DE METRIQUE.....	7
3.3. CONFIGURATION DES REGLES DE CODAGE.....	8
3.4. CONFIGURATION DES METRIQUES	9
4. VERIFICATION DU RESPECT DES REGLES DE CODAGE	11
4.1. LANCEMENT DE L'ANALYSE	11
4.2. AFFICHAGE DES RESULTATS.....	13
4.3. PARCOURS DES VIOLATIONS DETECTEES DANS LE CODE.....	14
4.4. EXPORT DES RESULTATS.....	15
5. MESURE DE LA COMPLEXITE.....	16
5.1. LANCEMENT DU CALCUL DES METRIQUES.....	16
5.2. AFFICHAGE DES RESULTATS.....	17
5.3. EXPORT DES RESULTATS.....	18
6. TABLES DES VIOLATIONS DETECTEES	19
6.1. REGLES COMMUNES	19
6.2. REGLES SPECIFIQUES.....	24
6.2.1. FORTRAN 77	24
6.2.2. FORTRAN 90	28
7. TABLES DES METRIQUES CALCULEES	38
8. MESSAGES UTILISATEUR	41
8.1. MESSAGES DES VIOLATIONS.....	41



Manuel Utilisateur i-Code CNES

Réf. : DCT/AQ /SO - 2014.0020026

Version : 1.2

Date : 16/11/2015

Page : 3/48

8.1.1. REGLES COMMUNES	42
8.1.2. FORTRAN 77	43
8.1.3. FORTRAN 90	45



	Manuel Utilisateur i-Code CNES	Réf. : DCT/AQ /SO - 2014.0020026 Version : 1.2 Date : 16/11/2015 Page : 4/48
---	---------------------------------------	---

Table des modifications

Version	Date	Objet
1.0	05/12/2014	Initialisation du document
1.1	23/01/2015	Prise en compte des modifications des standards Fortran 77, Fortran 90 et des règles communes
1.2	16/11/2015	Intégration logo, versions officielles des standards.

	<p align="center">Manuel Utilisateur i-Code CNES</p>	<p>Réf. : DCT/AQ /SO - 2014.0020026</p> <p>Version : 1.2</p> <p>Date : 16/11/2015</p> <p>Page : 5/48</p>
---	---	--

1. Introduction

Ce document constitue le manuel utilisateur de l'outil i-Code CNES. Il vise à décrire le fonctionnement et l'utilisation de ce plug-in eclipse (développé initialement pour la version Juno 4.2 d'eclipse, puis migré sur la version Mars).


Avant d'utiliser i-Code CNES, il est fortement conseillé de :



- prendre connaissance de l'environnement eclipse. La documentation d'eclipse est disponible sur le site en référence [R1].
- prendre connaissance des règles normatives du CNES sur les langages à analyser [R2] [R3] [R4]

2. Documents de référence

- [R1]. Documentation d'Eclipse :
- <http://help.eclipse.org/mars/index.jsp>
- [R2]. Règles pour l'utilisation du langage Fortran 77
- RNC-CNES-Q-HB-80-505 Version 7
- [R3]. Règles pour l'utilisation du langage Fortran 90
- RNC-CNES-Q-HB-80-517 Version 5
- [R4]. Règles communes pour l'utilisation des langages de programmation
- RNC-CNES-Q-HB-80-501 Version 5

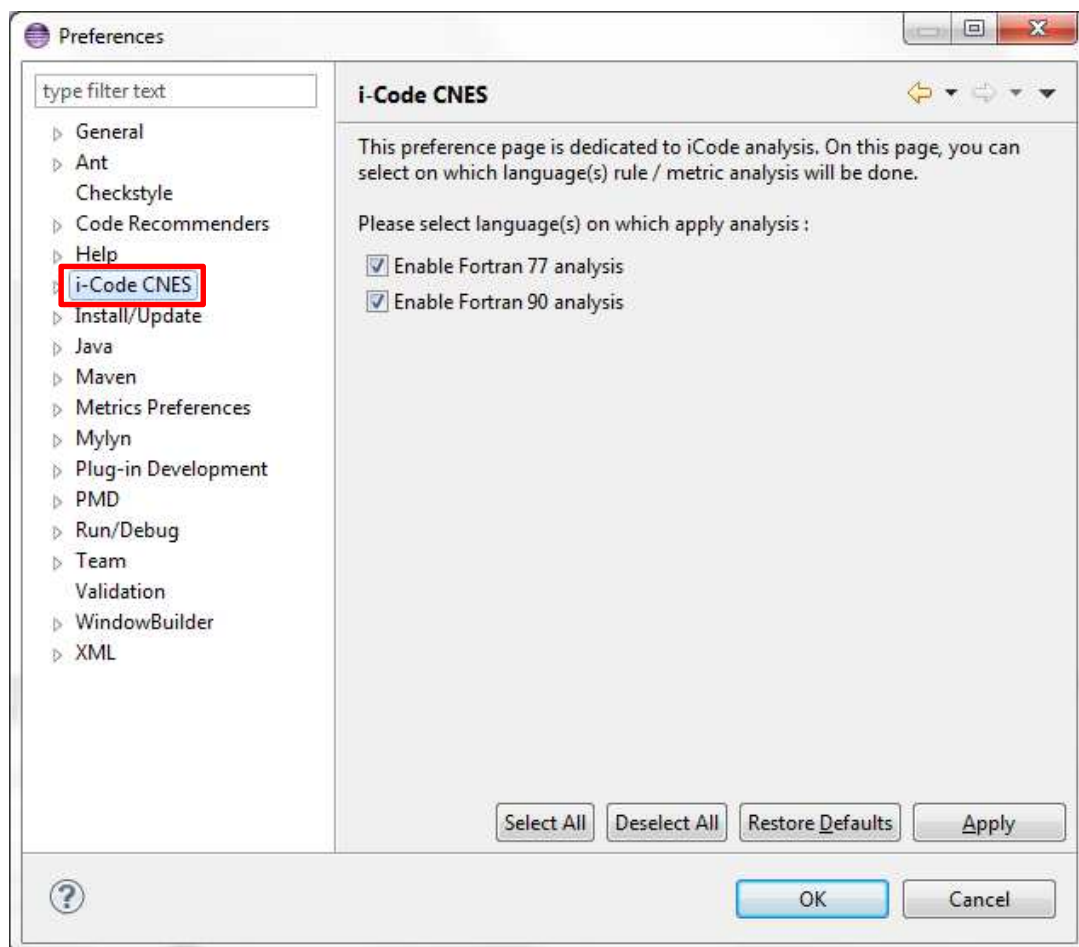
	<p align="center">Manuel Utilisateur i-Code CNES</p>	<p>Réf. : DCT/AQ /SO - 2014.0020026</p> <p>Version : 1.2</p> <p>Date : 16/11/2015</p> <p>Page : 6/48</p>
---	---	--

3. Configuration de l'outil

L'ensemble de la configuration de l'outil est accessible dans les pages de préférences d'eclipse.

Pour ouvrir les pages de préférence d'eclipse, cliquer sur [Windows > Préférences](#)

Dans la fenêtre des préférences, un item « i-Code CNES » est visible dans la partie gauche, c'est dans cette partie que la configuration peut être modifiée.




Page de préférence i-Code CNES

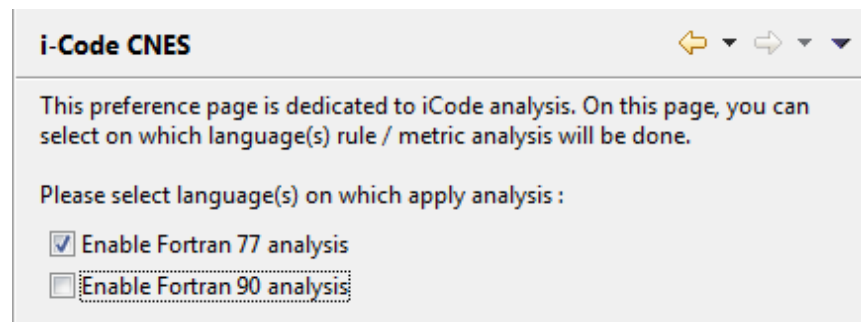
3.1. ACTIVATION / DESACTIVATION D'UN LANGAGE

i-Code CNES permet de vérifier le respect des règles de codage et de calculer les métriques de différents langages. La version 1 ne propose que Fortran77 et Fortran90, mais i-Code CNES intégrera, à termes, d'autres langages.

L'activation/désactivation d'un langage se fait sur la page de préférence principale i-Code CNES.

 i-Code CNES	Manuel Utilisateur i-Code CNES	Réf. : DCT/AQ /SO - 2014.0020026 Version : 1.2 Date : 16/11/2015 Page : 7/48
--	---------------------------------------	---

L'activation d'un langage rend accessible le menu correspondant à ce langage dans le menu i-Code CNES, nécessaire pour lancer une analyse. Elle rend également accessible les pages de préférence permettant de configurer les analyses de ce langage.



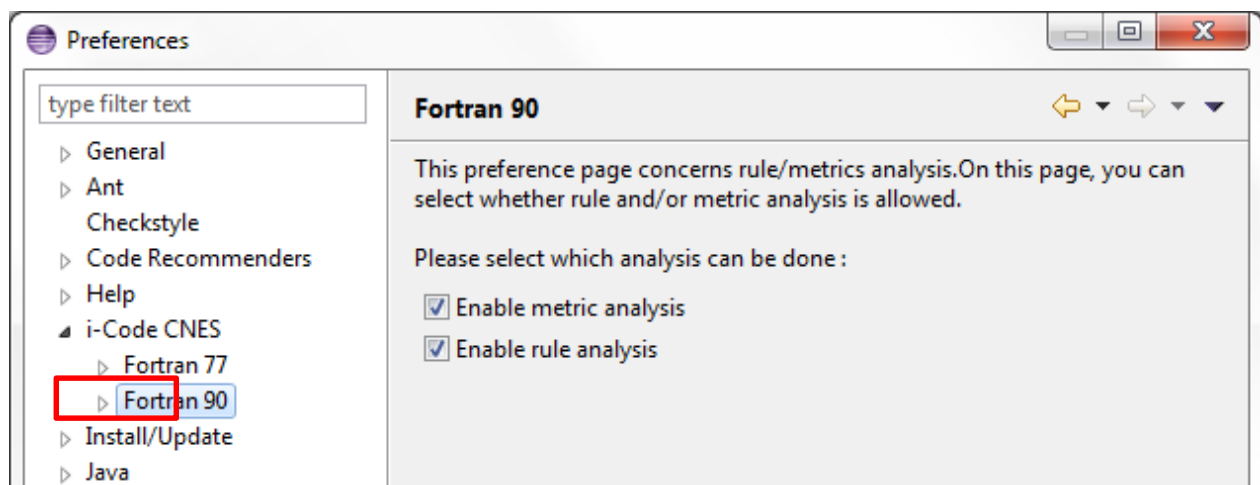
Page de préférence i-Code CNES

3.2. ACTIVATION / DESACTIVATION DE LA VERIFICATION DU RESPECT DES REGLES DE CODAGE ET DU CALCUL DE METRIQUE

Pour chaque langage, i-Code CNES permet de vérifier le respect des règles de codage et le calcul des métriques.

L'activation/désactivation de ces deux fonctionnalités se fait sur la page de préférence spécifique à chaque langage.

L'activation d'une fonctionnalité rend accessible le sous-menu correspondant dans le menu du langage concerné.



Page de préférence i-Code CNES Fortran 90

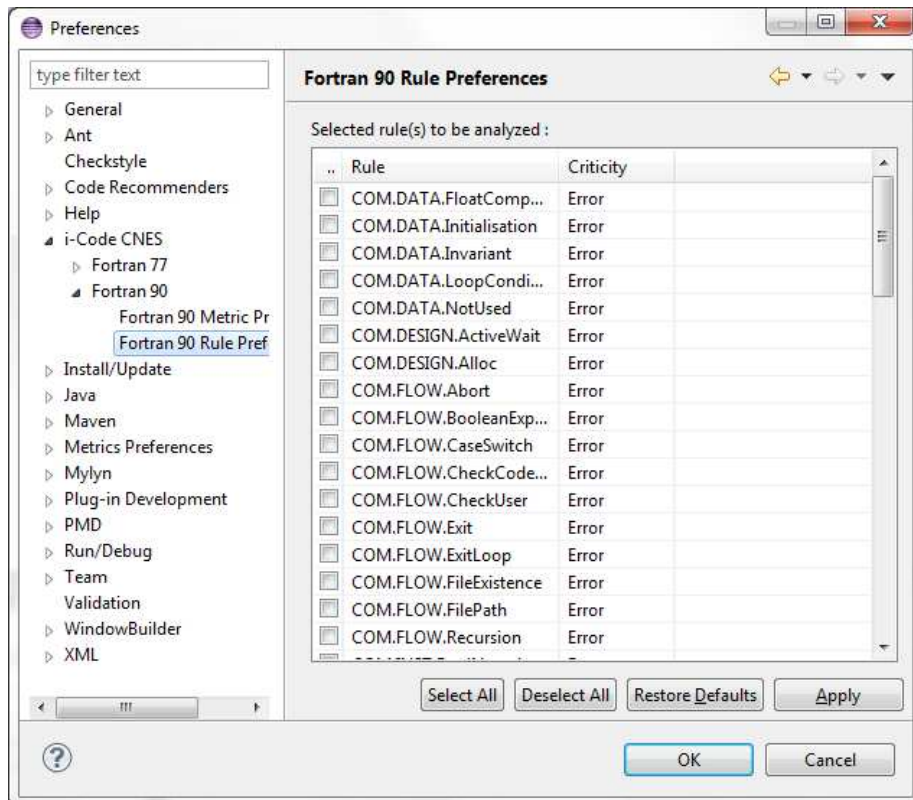


3.3. CONFIGURATION DES REGLES DE CODAGE

La configuration des règles de codage vérifiée par i-Code CNES se fait sur la page de préférence relative aux règles de codage du langage concerné.

La page de préférence relative aux règles d'un langage présente l'ensemble des règles que peut vérifier l'outil. Cette table présente la liste des règles ainsi que leur criticité (par défaut toutes les règles sont en criticité « error »).

Pour activer/désactiver une règle, il suffit de cocher/décocher la case correspondante dans la table.

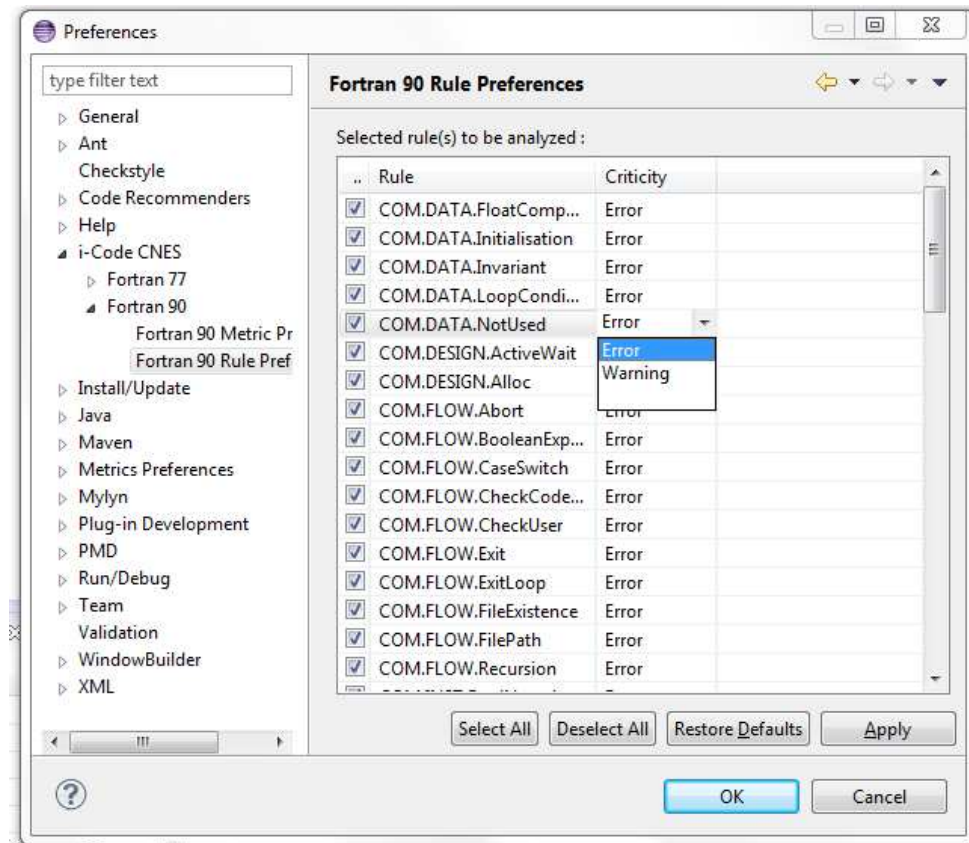


Page de préférence i-Code CNES des règles Fortran 90

- Le check de la règle est *active* : dans la prochaine analyse, cette règle sera analysée.
- Le check de la règle est *désactivé* : cette règle ne sera pas analysée à la prochaine analyse.
- Select all : toutes les règles seront activées.
- Deselect all : toutes les règles seront désactivées.
- Restore defaults : Rétablit les valeurs par défaut (toutes les règles activées)
- Apply : Valide et sauvegarde les modifications.



Pour modifier la criticité d'une règle de codage, il suffit de sélectionner la criticité choisie dans la table (« error » ou « warning »).

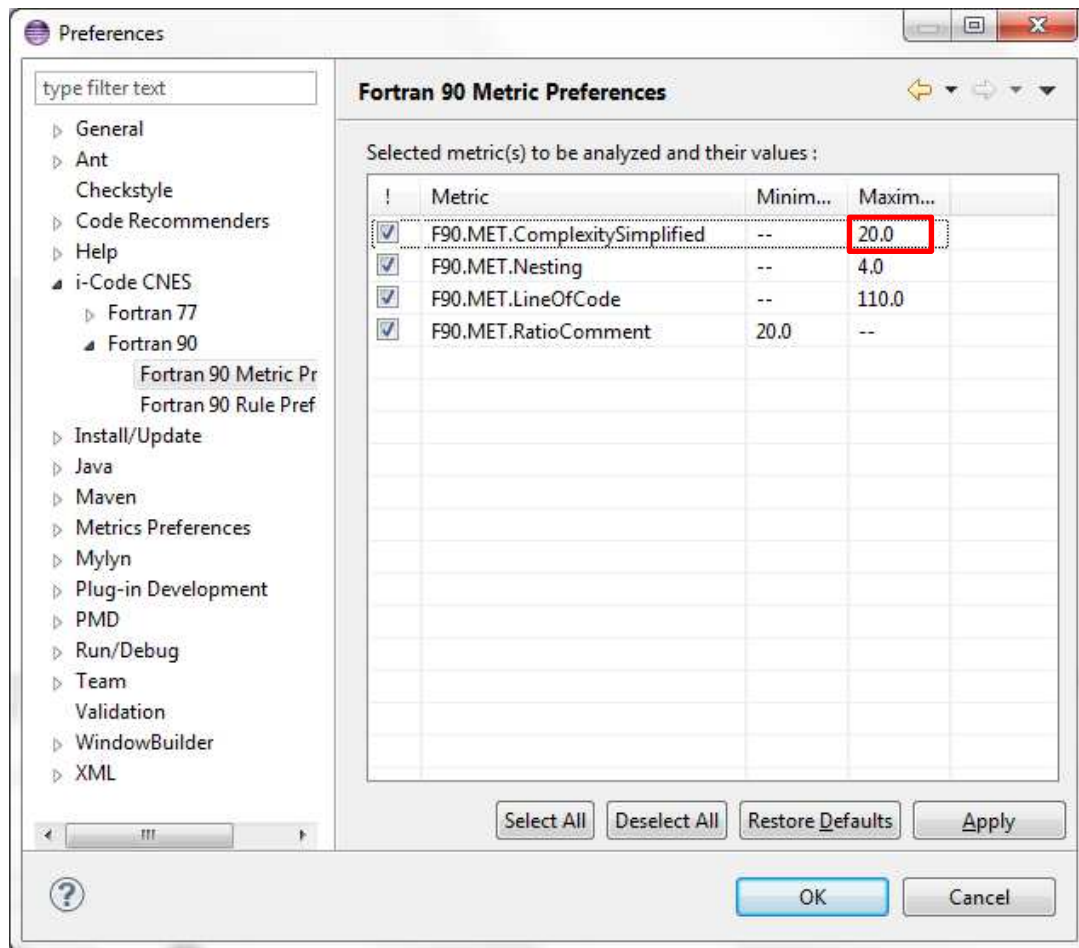


Page de préférence i-Code CNES des règles Fortran 90

3.4. CONFIGURATION DES METRIQUES

La configuration des métriques se fait sur la page de préférence relative aux métriques du langage concernée.

Pour activer/désactiver une métrique, il suffit de cocher/décocher la case correspondante dans la table.

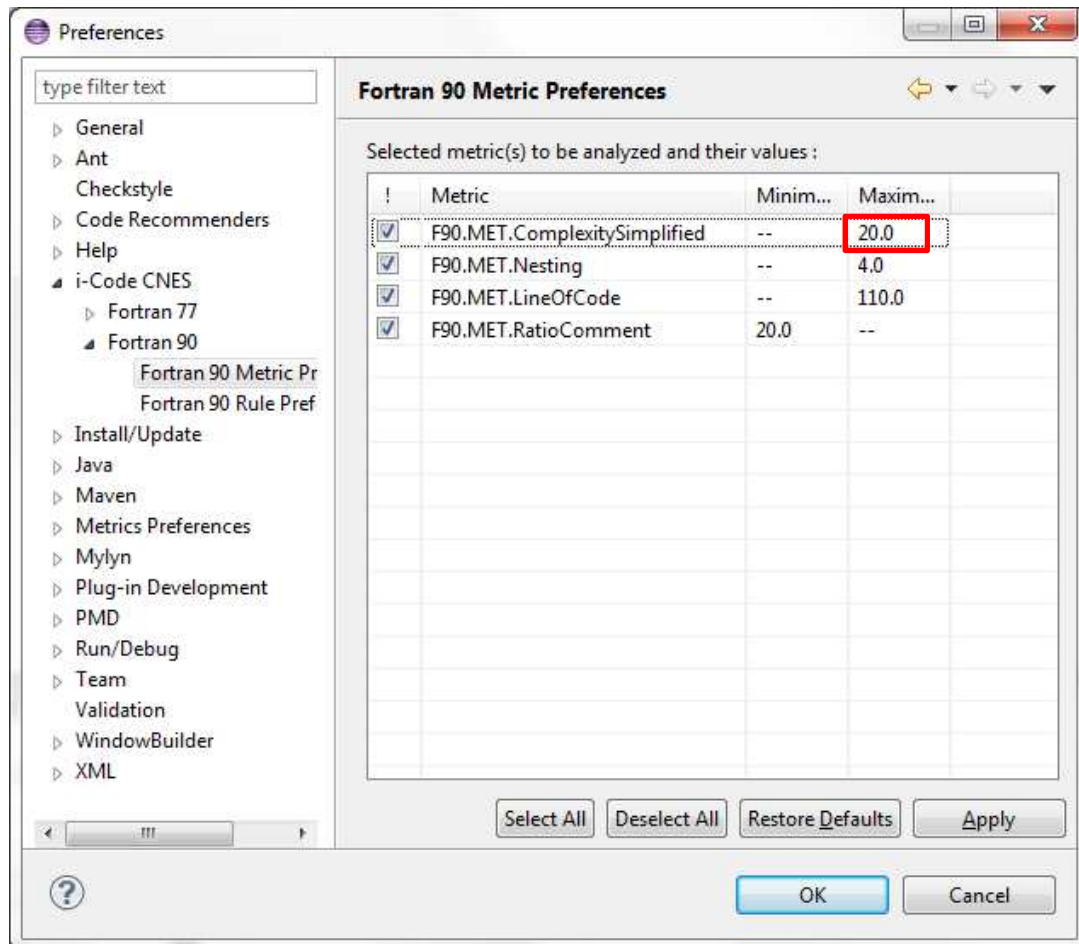


Page de préférence i-Code CNES des métriques Fortran 90

- Le calcul de la métrique est *actif* : dans la prochaine analyse, cette métrique sera calculée.
- Le calcul de la métrique est *désactivé* : dans la prochaine analyse, cette métrique ne sera pas calculée.
- *Select all* : toutes les métriques seront calculées.
- *Deselect all* : aucune métrique ne sera calculée.
- *Restore defaults* : Rétablit les valeurs par défaut (toutes les métriques activées)
- *Apply* : Valide et sauvegarde les modifications.



Pour modifier les seuils d'une métrique, il suffit de définir le seuil choisi dans la table.



Page de préférence i-Code CNES des métriques Fortran 90

4. Vérification du respect des règles de codage

i-Code CNES permet de vérifier certaines règles de codage des standards de Fortran 77 [R2], Fortran 90 [R3], et les règles communes [R4] sur les fichiers Fortran, de visualiser les résultats dans un tableau et d'accéder rapidement à la ligne incriminée.



Attention : il n'est pas nécessaire d'avoir du code qui compile pour utiliser i-Code CNES. Néanmoins, les erreurs syntaxiques peuvent provoquer des remontées de violations intempestives. Il est donc conseillé de s'assurer au préalable que le code compile.

4.1. LANCEMENT DE L'ANALYSE

Prérequis : L'ensemble des fichiers à analyser est disponible dans projet eclipse. La vue permettant de visualiser les résultats est vide (sinon utiliser le menu « Clear All »), sans quoi les résultats seront ajoutés aux résultats précédents.



Manuel Utilisateur i-Code CNES

Réf. : DCT/AQ /SO - 2014.0020026

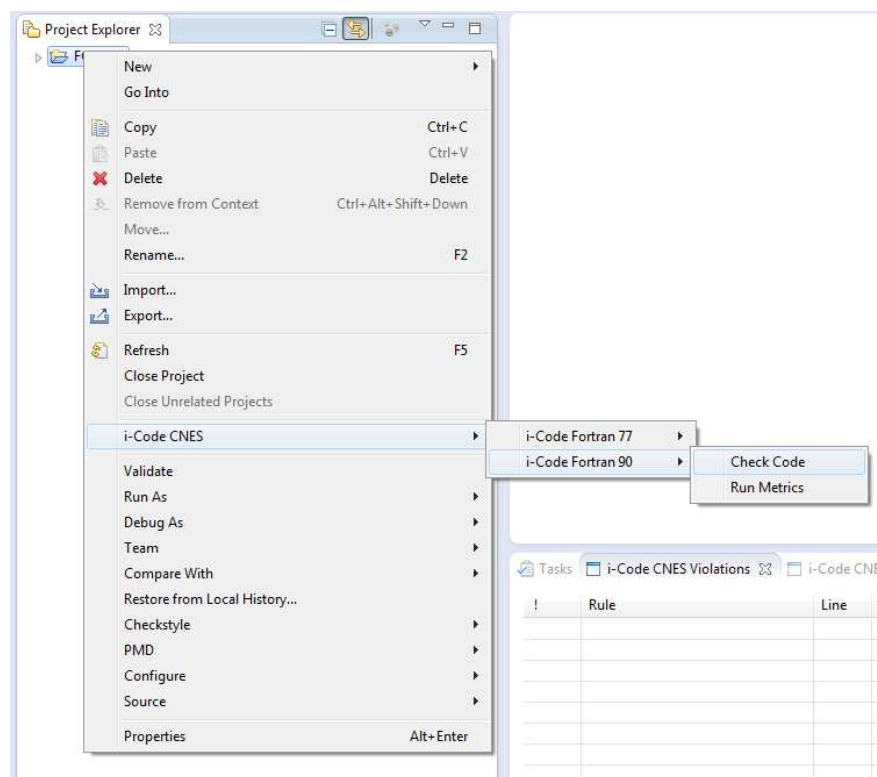
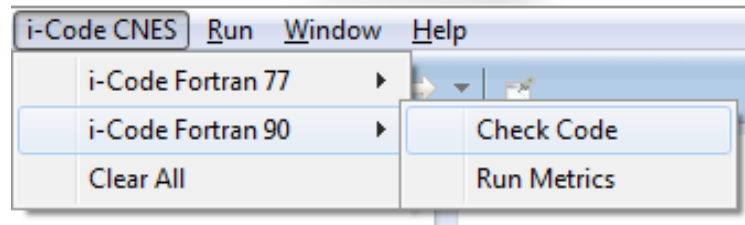
Version : 1.2

Date : 16/11/2015

Page : 12/48

Sélectionner le projet ou le fichier à analyser.

1. Lancer l'analyse via le menu i-Code (menu principal ou menu contextuel)



2. Une barre de progression apparaît. A la fin de l'analyse, les résultats s'affichent dans la vue « i-Code CNES Violations ».



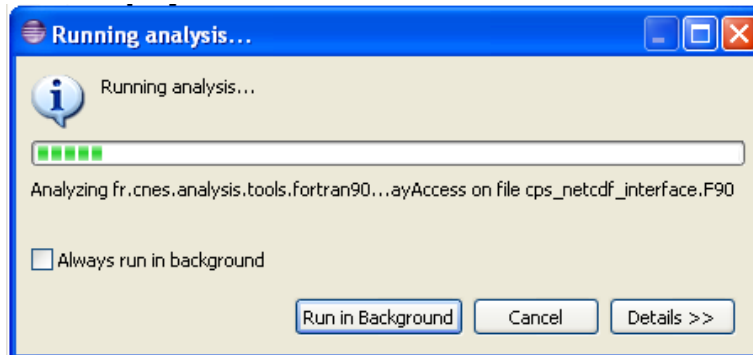
Manuel Utilisateur i-Code CNES

Réf. : DCT/AQ /SO - 2014.0020026

Version : 1.2

Date : 16/11/2015

Page : 13/48



4.2. AFFICHAGE DES RESULTATS

La vue *i-Code CNES Violations* présente les résultats de l'analyse pour chaque règle.

Tasks

i-Code CNES Violations

i-Code CNES Metrics

Search


!	Rule	Line	Num...	Message
▶	F90.DATA.Declaration	--	716	
▶	F90.ERR.Allocate	--	194	

La signification des colonnes de la vue est la suivante :

- *Rule* : nom de la règle qui provoque l'erreur
- *Line* : Numéro de ligne où est détectée l'erreur
- *Number of violations* : nombre total de violations
- *Message* : détails sur l'erreur (variable qui lance l'erreur, description, etc.)

L'ensemble des violations d'une même règle est présentée sous forme arborescente. Le premier niveau correspond au fichier présentant la violation, le deuxième niveau la méthode et la ligne.

Tasks					i-Code CNES Violations					i-Code CNES Metrics					Search				
!		Rule			Line		Number of violations			Message									
▶		F90.DATA.Declaration			--		716												
▶		F90.ERR.Allocate			--		194												
▶		adc_convert_module.f90			--		1												
▶		subroutine interf_adc_convert			246		--			The status of the ALLOCATE or DEALLOCATE instruction is not checked									
▶		adc_convert_type.f90			--		4												
▶		apf_performance_module.f90			--		1												
▶		apf_type.f90			--		20												
▶		bit_trimming_type.f90			--		4												
▶		ccm_computation_module.f90			--		4												
▶		ccm_type.f90			--		27												
▶		chromatism_computation_modul...			--		28												
▶		convolution_module.f90			--		57												
▶		cps_acces.f90			--		17												
▶		cps_acces77.f90			--		31												

	<p>Manuel Utilisateur i-Code CNES</p>	<p>Réf. : DCT/AQ /SO - 2014.0020026</p> <p>Version : 1.2</p> <p>Date : 16/11/2015</p> <p>Page : 14/48</p>
---	--	---

4.3. PARCOURS DES VIOLATIONS DETECTEES DANS LE CODE

La table des résultats présente le fichier et la ligne d'une violation. Il suffit d'ouvrir le fichier incriminé pour voir l'erreur dans le contexte. Pour cela, i-Code CNES permet d'ouvrir directement le fichier à la ligne correspondant à la violation par un double-clic sur la violation.

Un marqueur dans la colonne de gauche de l'éditeur permet d'identifier plus facilement les violations dans le code. Le survol d'un marqueur avec la souris permet d'afficher le nom de la règle dans un tooltip.

Dans La vue ProjectExplorer, le marqueur apparait sur les fichiers présentant des violations aux règles de codage



Manuel Utilisateur i-Code CNES

Réf. : DCT/AQ /SO - 2014.0020026

Version : 1.2

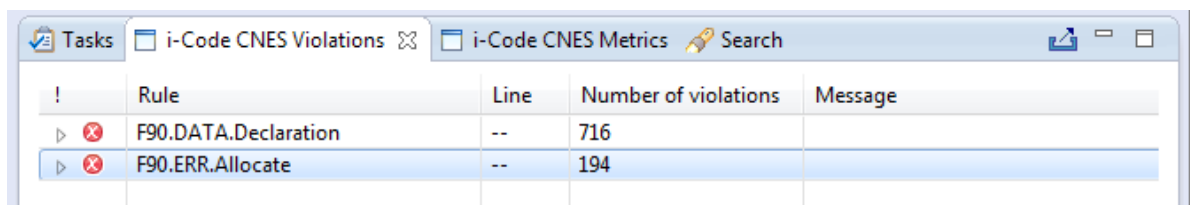
Date : 16/11/2015

Page : 15/48

4.4. EXPORT DES RESULTATS

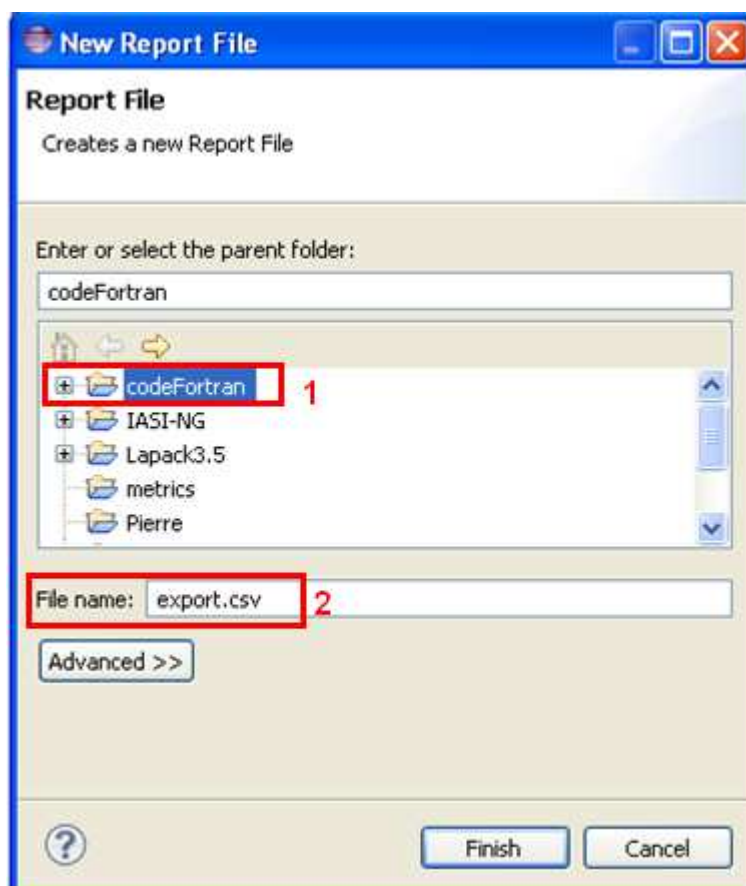
Les résultats de l'analyse peuvent être exportés dans un fichier au format csv.

1. Cliquer sur le bouton d'export en haut à droite de la vue « i-Code CNES Violations »



!	Rule	Line	Number of violations	Message
▶	F90.DATA.Declaration	--	716	
▶	F90.ERR.Allocate	--	194	

2. Indiquer le nom du fichier d'export et sa localisation.



3. Le fichier est créé dans le projet sélectionné.



5. Mesure de la complexité

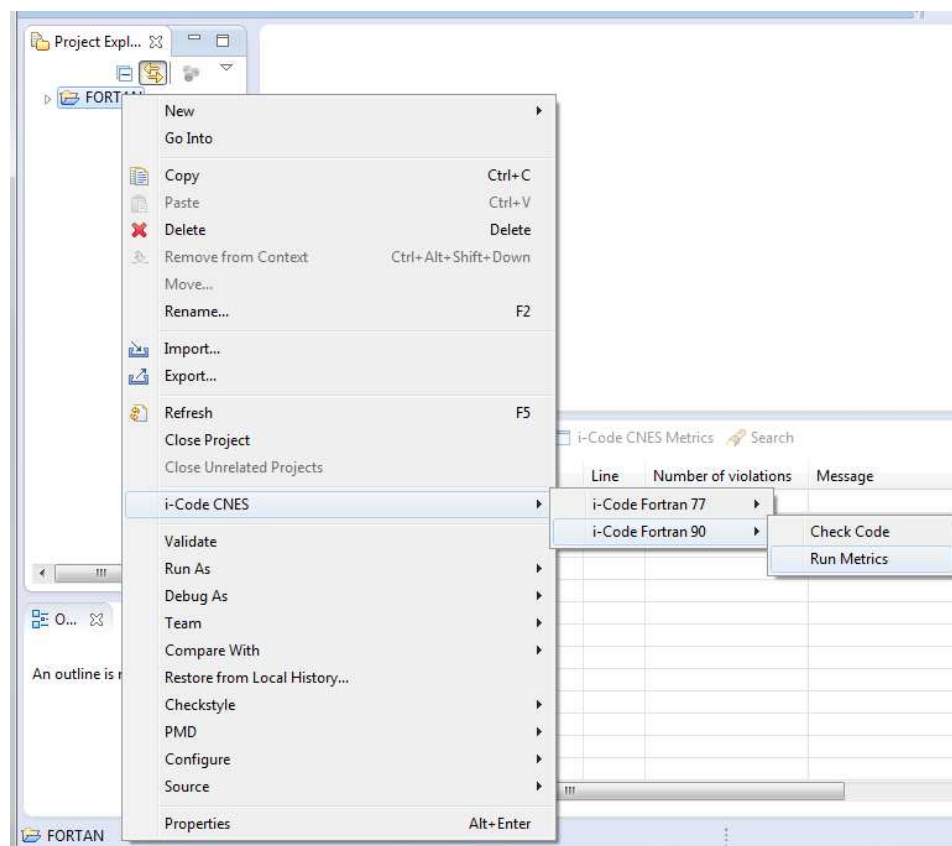
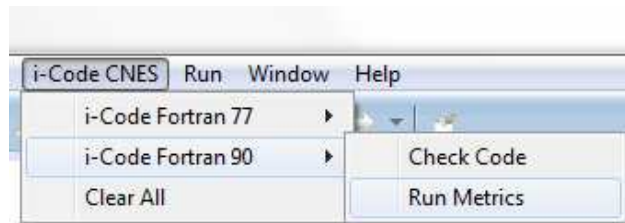


Attention : il n'est pas nécessaire d'avoir du code qui compile pour utiliser i-Code CNES. Néanmoins, les erreurs syntaxiques peuvent provoquer des remontées de violations intempestives. Il est donc conseillé de s'assurer au préalable que le code compile.

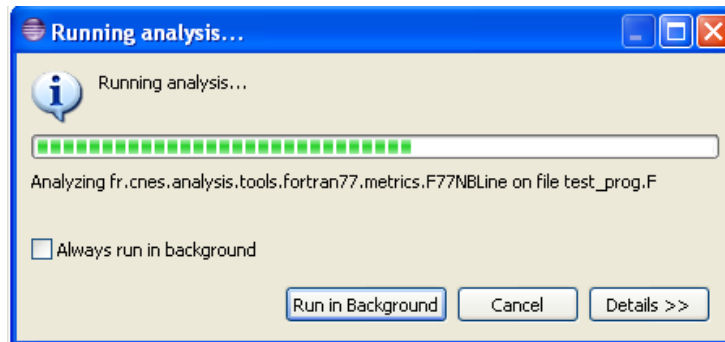
5.1. LANCEMENT DU CALCUL DES METRIQUES

Prérequis : L'ensemble des fichiers à analyser est disponible dans projet eclipse.

1. Sélectionner le projet ou le fichier à analyser.
2. Lancer l'analyse via le menu i-Code CNES (menu principal ou menu contextuel)



3. Une barre de progression apparait. A la fin de l'analyse, les résultats s'affichent dans la vue « i-Code CNES Metrics ».



5.2. AFFICHAGE DES RESULTATS

La vue *i-Code CNES Metrics* présente l'ensemble des métriques calculées.

Tasks i-Code CNES Violations i-Code CNES Metrics Search					
Metric	Total	Mean	Minimum	Maximum	Resource causing Minin
▶ F90.MET.ComplexitySimplified	--	--	--	--	aggregation_module.f90
▶ F90.MET.LineOfCode	108.0	108.0	108.0	108.0	aggregation_module.f90
▶ F90.MET.Nesting	--	--	--	--	aggregation_module.f90
▶ F90.MET.RatioComment	--	--	--	--	aggregation_module.f90

La signification des colonnes est la suivante :

- *Metric* : nom de la métrique qui est calculée
- *Total* : addition de toutes les valeurs de cette métrique pour tous les fichiers analysés
- *Mean* : moyenne des métriques pour les fichiers analysés
- *Minimum* : valeur minimum dans tous les fichiers
- *Maximum* : valeur maximum dans tous les fichiers
- *Resource causing minimum* : nom du fichier qui provoque le minimum
- *Resource causing maximum* : nom du fichier qui provoque le maximum

Les lignes sont colorées selon ses valeurs :

- Bleu : la valeur est comprise entre le minimum et maximum établi dans la page de préférences
- Rouge : la valeur est hors seuils



EXEMPLE : Le calcul de la métrique F90.MET.LineOfCode

!	Metric	Minim...	Maxim...
<input checked="" type="checkbox"/>	F90.MET.Com...	--	20.0
<input checked="" type="checkbox"/>	F90.MET.Nesti...	--	4.0
<input checked="" type="checkbox"/>	F90.MET.Line...	--	110.0
<input checked="" type="checkbox"/>	F90.MET.Ratio...	20.0	--

Les valeurs pour F90.MET.LineOfCode sont :

- Minimum : 0 lines
- Maximum : 110 lines



Metric	Total	Mean	Minimum	Maximum	Resource causing Mi
MSP_ACCES.F90	2699.0	2699.0	2699.0	2699.0	subroutine MSP_ferr
module procedure	2.0	2.0	2.0	2.0	
FUNCTION MSP_acc_sele	75.0	75.0	75.0	75.0	
FUNCTION MSP_acc_crea	77.0	77.0	77.0	77.0	
FUNCTION MSP_acc_sele	66.0	66.0	66.0	66.0	
function MSP_acc_charge	97.0	97.0	97.0	97.0	
subroutine MSP_acc_get_	63.0	63.0	63.0	63.0	
subroutine MSP_acc_get_	66.0	66.0	66.0	66.0	
subroutine MSP_acc_get_	62.0	62.0	62.0	62.0	
subroutine MSP_acc_get_	166.0	166.0	166.0	166.0	
subroutine MSP_acc_get_	95.0	95.0	95.0	95.0	
subroutine MSP_acc_get_	97.0	97.0	97.0	97.0	
subroutine MSP_acc_get_	95.0	95.0	95.0	95.0	

En conséquence les subroutines dépassant le seuil sont en rouges.

5.3. EXPORT DES RESULTATS

L'export des résultats en fichier csv, se fait de la même manière que l'export des résultats sur les règles. (Suivi point [3.4 Export des résultats](#) pour le détail)



Manuel Utilisateur i-Code CNES

Réf. : DCT/AQ /SO - 2014.0020026

Version : 1.2

Date : 16/11/2015

Page : 19/48

6. Tables des violations détectées

i-Code ne permet pas de vérifier toutes les règles. Les tableaux ci-dessous précisent ce qui est vérifié par l'outil. Pour les règles non couvertes ou partiellement couvertes, une vérification manuelle par échantillonnage est nécessaire.

6.1. REGLES COMMUNES

Règle	Vérification	Couverture du standard
COM.DATA.Array	Obligation d'utiliser les tableaux à deux dimensions de manière à avoir "ligne x colonne"	Non vérifiable automatiquement
COM.DATA.FloatCompare	Comparaison d'égalité/inégalité (.EQ., ==, .NE., /=) interdite entre des nombres réels (REAL, DOUBLE PRECISION ou COMPLEX).	Oui
COM.DATA.Initialisation	Les variables doivent être initialisées avant d'être utilisées. Quand une variable est utilisée dans le code, le programme vérifie qu'elle est initialisée (nom de la variable puis signe d'égalité), sinon, il renvoie une erreur.	Oui
COM.DATA.Invariant	Les données déclarées dans une subroutine, fonction, etc et jamais modifiées (pas d'occurrence de la variable puis signe d'égalité) doivent être définies comme constantes	Oui
COM.DATA.LoopCondition	Interdiction de modifier les données de condition de sortie des boucles à l'intérieur de celle-ci	Oui
COM.DATA.NotUsed	Fortran : Toute variable déclarée doit être utilisée, sinon une erreur est remontée.	Oui
COM.DATA.Using	Interdiction de réutiliser un objet local dans des traitements de type différent.	Non vérifiable automatiquement
COM.DESIGN.ActiveWait	Fortran : dans une boucle, les instructions SLEEP, WAIT et PAUSE sont interdites.	Oui
COM.DESIGN.Alloc	Fortran : L'allocation et la desallocation des ressources doit	PC ¹

¹ Les ressources vérifiées sont les blocs de mémoire, pas les fichiers. Si le développeur encapsule les allocations et desallocations dans les subroutines, l'application remonte une erreur.



Manuel Utilisateur i-Code CNES

Réf. : DCT/AQ /SO - 2014.0020026

Version : 1.2

Date : 16/11/2015

Page : 20/48

Règle	Vérification	Couverture du standard
	être dans le même niveau (fonction, sous-routine, ...). Chaque fois que le mot 'DEALLOCATE' est trouvé, le programme vérifie qu'il y a le mot 'ALLOCATE' et que les deux utilisent la même ressource.	
COM.FLOW.Abort	Fortran : Le mot STOP est interdit.	Oui
COM.FLOW.BooleanExpression	Dans une instruction conditionnelle (IF, DO) il n'est pas possible de définir plus de cinq expressions conditionnelles (AND, OR, NEQV, XOR, EQV, NOT, LT, <, LE, <=, GT, >, GE, >=, EQ, ==, NE, /=).	Oui
COM.FLOW.CaseSwitch	Fortran77 : Non Applicable	N/A
	Fortran90 : Obligation de finir l'instruction SWITCH avec DEFAULT, afin de traiter tous les cas possibles.	Oui
COM.FLOW.CheckArguments	Fortran : Obligation de contrôler les paramètres passés à un programme	Non
COM.FLOW.CheckCodeReturn	Fortran : Obligation de tester tous les retours de fonction	Oui
COM.FLOW.CheckUser	Fortran : Obligation de vérifier l'identité de l'utilisateur qui exécute un programme	Oui
COM.FLOW.Exit	Interdiction d'implémenter plusieurs points de sortie dans les fonctions, procédures ou méthodes.	Oui
COM.FLOW.ExitLoop	Interdiction d'implémenter plus d'une sortie dans les boucles.	Oui
COM.FLOW.FileExistence	Fortran : Avant d'ouvrir ou créer (mot clé OPEN, READ, WRITE) un fichier, doit apparaître l'instruction INQUIRE avec le flag EXIST sur le même fichier.	Oui
COM.FLOW.FilePath	Dans l'instruction OPEN, il est interdit d'utiliser directement le nom du fichier (fichier.txt). Le chemin d'accès doit être défini au travers d'une variable qui contient le chemin vers le fichier.	Oui
COM.FLOW.Recursion	Fortran77 : Non Applicable	N/A



Réf. : DCT/AQ /SO - 2014.0020026

Date : 16/11/2015

Page : 21/48

Règle	Vérification	Couverture du standard
	Fortran90 : Interdiction d'utiliser la récursivité. En Fortran, une fonction réursive est définie comme suit : RECURSIVE FUNCTION (params)	Oui
COM.INST.BoolNegation	La double négation est interdite sur les expressions booléennes. Les négations sont définies avec le mot .NOT. donc les expressions suivantes ne sont pas permis : .NOT. (.NOT. a) -> (a) .NOT. (a .AND. .NOT. b) -> .NOT. a .OR. b	Oui
COM.INST.Brace	Toute expression doit être parenthésée, ainsi le nombre de parenthèses ouvertes doit être supérieur ou égal au nombre d'opérateurs utilisés (+, -, *, /, **) a + b * c -> a + (b * c)	Oui
COM.INST.CodeComment	Fortran : Interdiction de commenter le code. Tout mot clé (ASSIGN, BACKSPACE, BLOCK DATA, CALL, ...) dans une ligne de commentaire est un erreur.. Le header (commentaires juste avant ou juste après de la déclaration de la fonction ou subroutine) peut contenir ces mots.	Oui
COM.INST.GOTO	Fortran: L'instruction GO TO est interdite.	Oui
COM.INST.Line	Fortran 77 : non applicable	N/A
	Fortran 90 et Shell : Chaque ligne doit contenir maximum une expression. En Fortran il est possible d'implémenter plusieurs instructions dans une ligne grâce au point-virgule. a = b + c ; d = e * f -> a = b + c d = e * f	Oui
COM.INST.LoopCondition	Dans une instruction de boucle, les comparaison d'égalité (.EQ., ==) ou de différence (!=, .NE.) sont interdites.	Oui
COM.NAME.Homonymy	Une variable doit d'avoir un nom unique. Chaque fois	Oui



Manuel Utilisateur i-Code CNES


Réf. : DCT/AQ /SO - 2014.0020026

Version : 1.2


Date : 16/11/2015

Page : 22/48

Règle	Vérification	Couverture du standard
	qu'une variable est trouvée, le programme vérifie que le nom de cette variable n'est pas déjà utilisé dans le programme.	
COM.PRES.Data	Obligation de commenter par une description détaillée les objets importants.	Non vérifiable automatiquement
COM.PRES.Indent	<p>Le code doit être indenté avec des espaces. Une ligne doit commencer à la même colonne que la ligne précédente. Après l'instruction DO, IF, WHILE, WHERE, SELECT et TYPE la ligne doit commencer dans une colonne supérieure, à l'exception de la fin de l'expression (dénotée par END). Exemple :</p> <pre>DO i = 2, nb somme = somme + x(i) IF (isnan(somme)) THEN print *, 'somme is a NaN' moy = -1.0 END IF END DO</pre>	Oui
COM.PRES.LengthLine	Une ligne de code doit contenir un maximum de 100 caractères. Le caractère 101 doit être écrit à la ligne suivante.	Oui
COM.PROJECT.Analyser	Obligation de passer un outil d'analyse statique sur tous les codes sources d'un projet.	Non vérifiable automatiquement
COM.PROJECT.CodeCloning	Interdiction de dupliquer / cloner du code.	Non
COM.PROJECT.Header	Obligation de définir et d'appliquer les entêtes/cartouches de chaque module et fonctions en début de projet	Oui

 i-Code CNES	Manuel Utilisateur i-Code CNES	Réf. : DCT/AQ /SO - 2014.0020026 Version : 1.2 Date : 16/11/2015 Page : 23/48
--	---------------------------------------	--

Règle	Vérification	Couverture du standard
COM.PROJECT.Warnings	Obligation d'afficher tous les warnings et de les corriger	Non vérifiable automatiquement
COM.TYPE.Expression	Fortran : Dans une expression (une expression est définie par un opérateur comme +, -, *, /, **), Les variables doivent être de même type : soit REAL, soit INTEGER, etc.	Oui

	Manuel Utilisateur i-Code CNES	Réf. : DCT/AQ /SO - 2014.0020026 Version : 1.2 Date : 16/11/2015 Page : 24/48
---	---------------------------------------	--

6.2. REGLES SPECIFIQUES

6.2.1. FORTRAN 77

Règle	Vérification	Couverture au standard
F77.BLOC.Common	Interdiction d'utiliser des COMMON blanc.	Oui
F77.BLOC.Else	Dans une instruction IF, le dernier ELSE IF doit toujours être suivi d'un ELSE.	Oui
F77.BLOC.File	Obligation d'utiliser les instructions OPEN et CLOSE pour accéder aux fichiers.	Non
F77.BLOC.Function	Obligation d'utiliser les parenthèses d'argument pour l'instruction FUNCTION, même s'il n'y a pas d'argument	Oui
F77.BLOC.Loop	Les boucles DO imbriquées doivent avoir des indicateurs de fermeture différents. Ce n'est pas possible de partager le label.	Oui
F77.DATA.Array	Obligation de déclarer explicitement les dimensions des tableaux. Par contre, est possible d'utiliser la notation * pour la dernière dimension mais toujours avec la justification d'un commentaire avant. A(*), A(4, *), A(4, *, *), A(4, 4, *), mais A(*, 4)	Oui
F77.DATA.Common	Obligation d'utiliser l'instruction INCLUDE pour déclarer les COMMON dans les unités de programmes qui les référencent.	Oui
F77.DATA.Double	Dans une initialisation de constante ou dans l'évaluation d'une expression arithmétique, l'utilisateur souhaite que cette constante soit évaluée en double précision, la présence d'un exposant double précision (lettre D) est obligatoire.	Oui
F77.DATA.Initialisation	Obligation d'initialiser toutes les variables avant leurs utilisations avec l'instruction DATA ou BLOCKDATA.	Oui
F77.DATA.IO	les unités logiques implicites définies par * sont interdites.	Oui



Manuel Utilisateur i-Code CNES

Réf. : DCT/AQ /SO - 2014.0020026

Version : 1.2

Date : 16/11/2015

Page : 25/48

Règle	Vérification	Couverture au standard
	READ (*,f) [iolist], READ f [,iolist], WRITE (*,f) [iolist], PRINT f [,iolist]	
F77.DATA.LoopDo	Obligation d'utiliser un type ENTIER comme paramètre de contrôle des boucles DO.	Oui
F77.DATA.Parameter	Interdiction d'utiliser des constantes, expression calculé ou appel de fonction comme paramètres de fonction. CALL function (3, x*y, f(z), var)	Oui
F77.ERR.OpenRead	Obligation des tester le statut de retour des instructions OPEN et READ, de préférence à l'aide du paramètre " IOSTAT = ", et vérifier le value de cette variable.	Oui
F77.INST.Assign	Interdiction d'utiliser l'instruction ASSIGN.	Oui
F77.INST.Dimension	Interdiction d'utiliser l'instruction DIMENSION.	Oui
F77.INST.Equivalence	Interdiction d'utiliser l'instruction EQUIVALENCE.	Oui
F77.INST.Function	Il faut utiliser l'instruction FUNCTION avec une déclaration explicite de type, à la définition de la fonction.	Oui
F77.INST.If	Interdiction d'utiliser le IF arithmétique : IF (Expression arithmétique) e1,e2,e3 Où les « eN » sont des étiquettes.	Oui
F77.INST.Include	Avec une instruction INCLUDE, le fichier inclus, ne peut pas inclure instructions exécutables (ASSIGN, GOTO, IF, ELSE, CONTINUE, STOP, PAUSE ; DO, READ, WRITE, PRINT, REWIND ;BACKSPACE, ENDFILE, OPEN, CLOSE, INQUIER, CALL, RETURN, END)	Oui
F77.INST.Pause	Interdiction d'utiliser l'instruction PAUSE.	Oui
F77.INST.Return	L'instruction RETURN(i) est interdite dans les sous-programmes.	Oui



Manuel Utilisateur i-Code CNES

Réf. : DCT/AQ /SO - 2014.0020026

Version : 1.2

Date : 16/11/2015

Page : 26/48

Règle	Vérification	Couverture au standard
F77.INST.Save	Interdiction d'utiliser l'instruction SAVE hormis pour des variables locales avec une justification par commentaire.	Oui
F77.MET.Line	Interdiction de dépasser 72 caractères par ligne.	Oui
F77.NAME.GenericIntrinsic	Obligation d'utiliser les noms génériques des fonctions intrinsèques.	Oui
F77.NAME.Intrinsic	Interdiction de réutiliser les noms des fonctions intrinsèques. Quand une fonction définie par le développeur a le même nom que une fonction intrinsèque (définie aux standards), l'application lance une erreur	Oui
F77.NAME.KeyWords	Interdiction de réutiliser les mots-clés u Fortran77 pour les variables	Oui
F77.NAME.Label	Restriction des étiquettes aux instructions FORMAT et CONTINUE.	Oui
F77.PROTO.Declaration	Obligation de déclarer les fonctions externes (lesquelles quisont pas dans le même fichier) par le mot EXTERNAL avant de leur utilisation.	Oui
F77.REF.IO	Obligation d'identifier les unités logiques par un nom symbolique. READ (5, *) NOMBRE -> READ (STDIN, *) NOMBRE	Oui
F77.REF.Open	Obligation de définir les paramètres FILE, STATUS et POSITION de l'instruction OPEN	Oui
F77.REF.Parameter	Interdiction de transmettre comme paramètre d'une subroutine els variables que son déjà dans un bloc COMMON accessible par la subroutine et le programme qui l'appel. COMMON /CONTROL/ A, B, C, D ... PROGRAM ESSAI CALL MY_SUB1 (A, B, C, D) ...	Oui



Manuel Utilisateur i-Code CNES

Réf. : DCT/AQ /SO - 2014.0020026

Version : 1.2

Date : 16/11/2015

Page : 27/48

Règle	Vérification	Couverture au standard
	<pre>END PROGRAM ESSAI SUBROUTINE MY_SUB1 (C_A, B, _C, C_D)</pre>	
F77.TYPE.Basic	Obligation d'utiliser les types standards (INTEGER, REAL, DOUBLE PRECISION, COMPLEX, LOGICAL, CHARACTER) uniquement. Autres types non standards seront considérées erreurs INTEGER*4 , LOGICAL*n	Oui
F77.TYPE.Hollerith	Les données et les constantes de type HOLLERITH sont interdites. Une donnée Hollerith est de la forme : numeroH par exemple 0.8H	Oui



Manuel Utilisateur i-Code CNES

Réf. : DCT/AQ /SO - 2014.0020026

Version : 1.2

Date : 16/11/2015

Page : 28/48

6.2.2. FORTRAN 90

Règle	Vérification	Couverture au standard
F90.BLOC.File	<p>Tout fichier ouvert doit être fermé. Le programme cherche l'instruction CLOSE pour chaque fichier ouvert en amont.</p> <pre>OPEN (unit = f_unit, ...) ... CLOSE (unit = f_unit, ...)</pre>	Oui
F90.DATA.Array	<p>La dimension du tableau doit être respectée. Pour la respecter, dans un appel au fonction, sera interdit de passer les parametres de limit inferieur et superieur à la table.</p> <p>Exemple :</p> <p>Si on la table suivante</p> <pre>integer, dimension(20) :: tab</pre> <pre>call subroutine_s1 (tab(1:20)) -> call subroutine_s1 (tab(:)) x = function_f1 (tab(a:b)) x = function_f1 (tab(:))</pre>	attente modification du standard
F90.DATA.ArrayAccess	<p>Dans un tableau d'indirection n'est pas possible de spécifier plusieurs fois le même élément.</p> <pre>Integer,dimension(3) :: a Integer.dimension(3) :: b a = (/ 1,1,3 /) b(a) = (/ 1,2,3 /)</pre>	Oui



Manuel Utilisateur i-Code CNES

Réf. : DCT/AQ /SO - 2014.0020026

Version : 1.2

Date : 16/11/2015

Page : 29/48

Règle	Vérification	Couverture au standard
F90.DATA.Constant	<p>Les constantes qui apparaissent dans plusieurs sous-programmes doivent être définies dans un module.</p> <p>Subroutine s1()</p> <p style="color: red;">Real PI = 3,141519</p> <p>End subroutine s1</p> <p style="text-align: right;">module precision</p> <p style="text-align: center;">-> real PI = 3.141519</p> <p>Function f1()</p> <p style="text-align: right;">end module precision</p> <p style="color: red;">Real PI = 3,141519</p> <p>End function</p>	Oui
F90.DATA.ConstantFloat	<p>Les constantes littérales numériques doivent être suivies par le paramètre de sous-type</p> <p>Integer,parameter :: DOUBLE=selected_real_kind(15)</p> <p>Real (DOUBLE) :: x = 0.1_DOUBLE</p>	Oui
F90.DATA.Declaration	<p>Obligation de placer la déclaration d'un objet avant toute référence à celui-ci ainsi que d'avoir l'instruction IMPLICIT NONE en début de chaque module</p>	Oui
F90.DATA.Float	<p>Est interdit d'utiliser le format * en sortie (instruction WRITE) pour les nombres flottants</p> <p>Real :: X</p> <p>Write (std_out, *), x</p>	Oui
F90.DATA.Parameter	<p>Les fonctions intrinsèques SELECTED_REAL_KIND et SELECTED_INT_KIND doivent être regroupées dans un module.</p> <p>MODULE precision</p> <p>Integer, parameter :: DOUBLE = SELECTED_REAL_KIND(15)</p>	Oui



Manuel Utilisateur i-Code CNES

Réf. : DCT/AQ /SO - 2014.0020026

Version : 1.2

Date : 16/11/2015

Page : 30/48

Règle	Vérification	Couverture au standard
	END MODULE	
F90.DESIGN.Free	Obligation de libérer la mémoire allouée dans le même niveau conceptuel.	Non
F90.DESIGN.Include	L'include d'un fichier est interdit. Si l'INCLUDE contient un fichier écrit en F90, le programme retourne une erreur. INCLUDE 'file_to_include.F90'	Oui
F90.DESIGN.Interface	Le contenu des modules doit être limité aux clauses USE, PRIVATE et PUBLIC MODULE interface_syslog Implicit none PRIVATE Interface Subroutine f_syslog(cdata) USE message_syslog Type(opendata_type) End subroutine End interface PUBLIC f_syslog End module interface_syslog	Oui
F90.DESIGN.IO	Le nombre d'unité dans une fonction OPEN doit dépendre d'une autre fonction ou un tableau.	PC ³

³ La règle demande de vérifier ces trois cas : 1) Des primitives d'allocation et de libération de numéros d'unité. 2) Une primitive de réservation d'un numéro d'unité donné, pour permettre l'utilisation de sous-programmes Fortran 77 utilisant des numéros fixés. 3) Des constantes nommées pour l'entrée et la sortie standard. Notre vérification inclue les options 2 et 3.



Manuel Utilisateur i-Code CNES

Réf. : DCT/AQ /SO - 2014.0020026

Version : 1.2

Date : 16/11/2015

Page : 31/48

Règle	Vérification	Couverture au standard
	<pre>Integer :: f_unit = 15 integer :: f_unit OPEN (UNIT = f_unit, ...) -> f_unit = getNumber() OPEN(UNIT = f_unit, ...)</pre>	
F90.DESIGN.Obsolete	<p>Cette règle vérifie les clauses suivantes :</p> <ul style="list-style-type: none">- Ne pas utiliser le GOTO calculé.- Ne pas utiliser la syntaxe : CHARACTER*N- Ne pas utiliser le IF arithmétique- Dans une boucle DO, ne pas utiliser de variables réelles ni comme indice, ni pour les bornes de l'intervalle de contrôle, ni pour le pas d'incrément.- Ne pas utiliser de terminaison de boucle DO autre que END DO ou CONTINUE.- Ne pas faire de branchements sur ENDIF.- Ne pas utiliser l'instruction PAUSE.- Ne pas utiliser l'instruction GOTO assigné.- Ne pas utiliser l'affectation d'étiquette de FORMAT- Ne pas utiliser le descripteur H (Hollerith) dans les formats.	Oui
F90.ERR.Allocate	<p>L'allocation (ALLOCATE) et libération (DEALLOCATE) doit contenir le paramètre STAT. Ensuite, le value du STAT doit être testé après l'instruction.</p> <pre>ALLOCATE(x, STAT = iom) IF (iom > 0) THEN ...</pre>	Oui
F90.ERR.OpenRead	<p>Les instructions OPEN et READ qui travaille avec des fichiers doivent contenir le paramètre IOSTAT, et vérifier le value de cette variable. Pour vérifier ça, l'instruction OPEN doit contenir l'attribut FILE. Une</p>	Oui



Manuel Utilisateur i-Code CNES

Réf. : DCT/AQ /SO - 2014.0020026

Version : 1.2

Date : 16/11/2015

Page : 32/48

Règle	Vérification	Couverture au standard
	<p>exemple qui ne respecte pas la règle est :</p> <p>OPEN (UNIT = FILE_UNIT, FILE = C_ARG)</p> <p>Parce que il n'y a pas d'attribut IOSTAT, et le value de IOSTAT n'est pas testé. Le même exemple correct sera :</p> <p>OPEN (UNIT = FILE_UNIT, FILE = C_ARG, IOSTAT=IOS)</p> <p>IF (IOS .NE. 0) ...</p> <p>Pour l'instruction READ, comme l'attribut FILE n'existe pas, la vérification sera sur ces reads que le UNIT désigne une unité logique qui a été ouverte par un OPEN avant a le code. En continuation avec l'antérieur exemple :</p> <p>READ (*, *) //pas vérifiable, lecture du clavier</p> <p>READ (UNIT = FILE_UNIT, FMT = 9011, IOSTAT = IOS) // verifiable</p> <p>IF (IOS .LT. 0) ...</p>	
F90.INST.Associated	<p>Entre le mot ASSOCIATED et la déclaration il faut avoir l'instruction NULLIFY.</p> <pre>graph TD decl[déclaration] --> eq1[=>] decl --> nullify[nullify] decl --> assoc1[associated] eq1 --> assoc2[associated] nullify --> eq2[=>] nullify --> assoc3[associated] eq2 --> assoc4[associated] assoc1 --- V1[V] assoc3 --- V2[V] assoc4 --- V3[V] assoc1 --- X[X]</pre> <p>Exemple :</p> <p>Real, pointer :: ptr</p>	Oui



Manuel Utilisateur i-Code CNES

Réf. : DCT/AQ /SO - 2014.0020026

Version : 1.2

Date : 16/11/2015

Page : 33/48

Règle	Vérification	Couverture au standard
	... NULLIFY (ptr) ASSOCIATED(ptr)	
F90.INST.Entry	L'instruction ENTRY est interdite Subroutine s1 ENTRY rien	Oui
F90.INST.Equivalence	Interdiction d'utiliser l'instruction EQUIVALENCE. INTEGER total (3,2) INTEGER sum (6) EQUIVALENCE (sum, total)	Oui
F90.INST.If	L'instruction IF logique est interdite quand est suivi par un mt autre que EXIT, CYCLE, GOTO et RETURN. IF (x == 0) THEN GO TO 1000 End IF	Oui
F90.INST.Intent	Chaque paramètre des sous-programmes doit avoir le mot clé INTENT à sa déclaration. Function f1(x, y, z) Integer, INTENT (IN) :: x Integer, INTENT (IN) :: y Integer, INTENT (OUT) :: z	Oui
F90.INST.Nullify	Après une desallocation il y a obligation d'utiliser l'instruction NULLIFY sur la même unité logique. DEALLOCATE (C, stat = iom)	



Manuel Utilisateur i-Code CNES

Réf. : DCT/AQ /SO - 2014.0020026

Version : 1.2

Date : 16/11/2015

Page : 34/48

Règle	Vérification	Couverture au standard
	NULLIFY (C)	
F90.INST.Only	Interdiction d'utiliser le mot clé ONLY sans commentaire avant qui explique son utilisation. Use mes_fonctions_intrinseques , ONLY :: getuid -> my_getuid	Oui
F90.INST.Operator	Ne pas utiliser la notation ancienne pour les opérateurs relationnels. Substituer .EQ., .NE., .LT., .LE., .GT., .GE. pour ==, /=, <, <=, >, >=	Oui
F90.INST.Pointer	Le POINTER est interdit à exception des cas suivantes : - pour créer des structures de données complexes (i.e. liste de chaînes, arbre, etc.) ; - pour manipuler des références à des tableaux (référence à un tableau alloué dans un sous-programme, recopie par échange de pointeurs,...) et à des parties de tableaux; - pour utiliser de l'allocation dynamique dans les composants de types dérivés. Le programme lance, donc, une erreur quand il trouve l'attribut POINTER et il référence une variable simple. real, pointer :: ppi	Oui
F90.NAME.GenericIntrinsic	Ne pas se servir des fonctions intrinsèques spécifiques (INT,IFIX,IDINT,REAL,FLOAT,SNGL,ICHAR,CHAR,AINT,DINT,ANINT,DNINT,NINT,IDNINT,IABS,ABS,DABS,CABS,MOD,AMOD,DMOD,ISIGN,SIGN,DSIGN,IDIM,DIM,DDIM,DPROD,MAX0,AMAX1,DMAX1,AMAX0,MAX1,MIN0,AMIN1,DMIN1,AMIN0,MIN1,AIMAG,CONJG,SQRT,DSQRT,CSQRT,EXP,DEXP,CEXP,ALOG,DLOG,CLOG,ALOG10,DLOG10,SIN,DSIN,CSIN,COS,DCOS,CCOS,TAN,DTAN,ASIN,DASIN,ACOS,DACOS,ATAN,DATAN,ATAN2,DATAN2,SINH,DSINH,COSH,DCOSH,TANH,DTANH), utiliser les génériques (INT,REAL,AINT,ANINT,NINT,ABS,MOD,SIGN,DIM,MAX,MIN,SQRT,EXP,LOG,LOG10,SIN,COS,TAN,ASIN,ACOS,ATAN,ATAN2,SINH,COSH,TANH).	Oui



Manuel Utilisateur i-Code CNES

Réf. : DCT/AQ /SO - 2014.0020026

Version : 1.2

Date : 16/11/2015

Page : 35/48

Règle	Vérification	Couverture au standard
	<p>resultat = AMOD (argument, diviseur)</p> <p>resultat = MOD (argument, diviseur)</p>	
F90.NAME.KeyWords	<p>Les variables du code ne peuvent être nommées comme les mots clés en Fortran (ALLOCATABLE, ALLOCATE ; ASSIGN, BACKSPACE, ...). De plus les noms de fonctions doivent être différents des fonctions intrinsèques (ABS, ACHAR, ACOS, ...)</p> <p>integer, parameter :: DATA</p> <p>integer, parameter :: MY_DATA</p>	Oui
F90.PROTO.Overload	La surcharge des opérateurs est interdite. Celle-ci est définie par l'instruction INTERFACE OPERATOR (symbole) et ensuite par leur utilisation.	Oui
F90.REF.ARRAY	<p>Dans une expression, quand on veut représenter l'utilisation total d'un tableau, il est obligatoire de se servir de la notation (:)</p> <p>Y = A*X + B</p> <p>Y(:) = A(:)*X + B où Y et A sont des tableaux</p>	Oui
F90.REF.Interface	<p>Le sous-programme appelé doit être visible</p> <p>subroutine Pas_1 subroutine Pas_1</p> <p>call Mouv(3.0,resul) call Mouv(3.0,resul)</p> <p>end subroutine Pas_1 contains</p> <p>subroutine Mouv(oper0, resul) -> subroutine Mouv(oper0, resul)</p> <p>... ...</p> <p>end subroutine Mouv end subroutine Mouv</p> <p> end subroutine Pas_1</p>	Oui
F90.REF.Label	L'END doit être suivi par le type (FUNCTION, SUBROUTINE, ...) et le nom	Oui



Manuel Utilisateur i-Code CNES

Réf. : DCT/AQ /SO - 2014.0020026

Version : 1.2

Date : 16/11/2015

Page : 36/48

Règle	Vérification	Couverture au standard
	<pre> function f ... end function </pre> <p>-></p> <pre> function f ... end function f </pre>	
F90.REF.Open	<p>Toute instruction OPEN doit avoir les paramètres FILE, STATUS, IOSTAT et POSITION</p> <pre> OPEN (UNIT=f_unit, FILE=c_args, STATUS='old', POSITION='rewind', IOSTAT=ios) </pre>	Oui
F90.REF.Variable	<p>Une même variable doit être référencée sous le même nom dans un sous-programme.</p> <pre> Call incr(i) ... subroutine incr(j) i = i + 1 end subroutine incr </pre> <p>-></p> <pre> call incr(i) ... subroutine incr(j) j = j + 1 end subroutine incr </pre>	Oui
F90.TYPE.Derivate	<p>Toute déclaration de type doit être dans un module.</p>	Oui
F90.TYPE.Integer	<p>Les paramètres INTEGER doivent être suivis par l'expression SELECTED_INT_KIND.</p> <pre> Integer, parameter :: LONG </pre> <pre> integer, parameter :: LONG = SELECTED_INT_KIND(5) </pre>	Oui
F90.TYPE.Real	<p>Les paramètres REAL doivent être suivis par l'expression SELECTED_REAL_KIND.</p> <pre> Real, parameter :: LONG </pre>	Oui



Manuel Utilisateur i-Code CNES

Réf. : DCT/AQ /SO - 2014.0020026

Version : 1.2

Date : 16/11/2015

Page : 37/48

Règle	Vérification	Couverture au standard
	<code>real, parameter :: LONG = SELECTED_REAL_KIND(5)</code>	



Manuel Utilisateur i-Code CNES

Réf. : DCT/AQ /SO - 2014.0020026

Version : 1.2

Date : 16/11/2015

Page : 38/48

7. Tables des métriques calculées

Dans le cas des métriques, on identifie les vérifications faites pour un fichier et celles faites pour une fonction – dans le cas où elles sont effectuées

Métrique	Vérification sur fichier
COM.MET.Nesting	<p>Nombre maximum de niveaux d'imbrications de if, switch,while, for, ... dans une fonction/méthode. Ce nombre est 0 s'ily a aucune imbrication..</p> <pre>SUBROUTINE changer_coordonnees(Deplacement, NbPoints, Points) ! ! --- Cette routine effectue une modification de coordonnees sur ! --- en appliquant un déplacement sur les 3 axes x, y et z ! IMPLICIT NONE INTEGER :: c ! colonne INTEGER :: l ! ligne INTEGER, parameter :: NbDim = 3 INTEGER, intent(in) :: NbPoints DOUBLE PRECISION, intent(inout), dimension(NbDim,NbPoints) DOUBLE PRECISION, intent(in), dimension(NbDim) :: Deplacement ! On applique a chaque point une valeur de déplacement selon les 1 do c=1, NbDim, 1 2 do l=1, NbPoints, 1 Points(c, l) = Points(c, l) + Deplacement(c) end do end do END SUBROUTINE</pre> <p>NB.Imbric = 2</p>
COM.MET.ComplexitySimplified	<p>C'est le nombre de décisions du code (nb if, case, while, catch...).</p>



Manuel Utilisateur i-Code CNES

Réf. : DCT/AQ /SO - 2014.0020026

Version : 1.2

Date : 16/11/2015

Page : 39/48

```
SUBROUTINE changer_coordonnees(Deplacement, NbPoints, Points)
!
! --- Cette routine effectue une modification de coordonnees sur
! --- en appliquant un déplacement sur les 3 axes x, y et z
!
    IMPLICIT NONE

    INTEGER :: c          ! colonne
    INTEGER :: l          ! ligne

    INTEGER, parameter :: NbDim = 3
    INTEGER, intent(in) :: NbPoints

    DOUBLE PRECISION, intent(inout), dimension(NbDim, NbPoints)
    DOUBLE PRECISION, intent(in), dimension(NbDim) :: Deplacement

    ! On applique a chaque point une valeur de déplacement selon les
1    do c=1, NbDim, 1
2        do l=1, NbPoints, 1
            Points(c, l) = Points(c, l) + Deplacement(c)
        end do
    end do

END SUBROUTINE
```

NB.Cycomatique = 2

COM.MET.LineOfCode

C'est le nombre total des lignes dans du code source du composant logiciel sauf les lignes vides et les lignes de commentaires



Manuel Utilisateur i-Code CNES

Réf. : DCT/AQ /SO - 2014.0020026

Version : 1.2

Date : 16/11/2015

Page : 40/48

```
1  SUBROUTINE changer_coordonnees(Deplacement, NbPoints, Points)
!
! --- Cette routine effectue une modification de coordonnees sur
! --- en appliquant un déplacement sur les 3 axes x, y et z
2  !
      IMPLICIT NONE
3
      INTEGER :: c          ! colonne
4      INTEGER :: l          ! ligne
5
      INTEGER, parameter :: NbDim = 3
6      INTEGER, intent(in) :: NbPoints
7
      DOUBLE PRECISION, intent(inout), dimension(NbDim,NbPoints)
8      DOUBLE PRECISION, intent(in), dimension(NbDim) :: Deplacement
9
      ! On applique a chaque point une valeur de déplacement selon les
10     do c=1, NbDim, 1
11         do l=1, NbPoints, 1
12             Points(c, l) = Points(c, l) + Deplacement(c)
13         end do
14     end do
15 END SUBROUTINE
```

NB.Line = 15

COM.MET.RatioComment

C'est la proportion de commentaires dans le code source du composant logiciel. L'entête se prend en compte dans le calcul de le taux de commentaire (tant si elle est avant ou après la définition).



```
1  SUBROUTINE changer_coordonnees(Deplacement, NbPoints, Points)
1  !
2  ! --- Cette routine effectue une modification de coordonnees sur
3  ! --- en appliquant un deplacement sur les 3 axes x, y et z
2  !
      IMPLICIT NONE
3      INTEGER :: c          ! colonne
4      INTEGER :: l          ! ligne
5
6      INTEGER, parameter :: NbDim = 3
6      INTEGER, intent(in) :: NbPoints
7
8      DOUBLE PRECISION, intent(inout), dimension(NbDim,NbPoints)
8      DOUBLE PRECISION, intent(in), dimension(NbDim) :: Deplacement
5  ! On applique a chaque point une valeur de deplacement selon les
9  do c=1, NbDim, 1
10 do l=1, NbPoints, 1
11     Points(c, l) = Points(c, l) + Deplacement(c)
12 end do
13 end do
14
15] END SUBROUTINE
```

RATE.Comment = 5 / 15 = 0.33

8. Messages utilisateur

Il y a deux types de messages différents dans l'outil : les messages qui correspondent à les violations et les messages d'informations dans une fenêtre émergente.

8.1. MESSAGES DES VIOLATIONS

Les messages qui concernent les violations sont montrées dans la vue *i-Code CNES violations* une fois que l'analyse est lancé. Ces messages donne un peu d'information sur l'erreur remonté :variable qui provoque l'erreur, petite explication de pourquoi cette erreur.



Manuel Utilisateur i-Code CNES

Réf. : DCT/AQ /SO - 2014.0020026

Version : 1.2

Date : 16/11/2015

Page : 42/48

8.1.1. REGLES COMMUNES

Règle	Message
COM.DATA.DeclarationOrder	The parameters are not defined in the right order. The order shall be: in, in/out, out.
COM.DATA.FloatCompare	It's not allowed to compare float variables(" <i>variable</i> ") with equality.
COM.DATA.Initialisation	The variable " <i>variable</i> " is used before being initialized.
COM.DATA.Invariant	The variable " <i>variable</i> " must be defined as constant.
COM.DATA.LoopCondition	The variable " <i>variable</i> " is modified inside the loop.
COM.DATA.NotUsed	The variable " <i>variable</i> " is declared and not used
COM.DESIGN.ActiveWait	This process contains an active wait.
COM.DESIGN.Alloc	The resource named " <i>variable</i> " has not been allocated and deallocated in the same algorithmic level.
COM.FLOW.Abort	The keyword STOP is not allowed.
COM.FLOW.BooleanExpression	Using more than five conditions in an expression is not allowed.
COM.FLOW.CaseSwitch	A DEFAULT case is needed in a switch case instruction.
COM.FLOW.CheckCodeReturn	The return code of the function " <i>function</i> " is not checked.
COM.FLOW.CheckUser	The user identity is not verified in the main program.
COM.FLOW.Exit	There is more than one exit in the function.
COM.FLOW.ExitLoop	There is more than one exit in the loop.
COM.FLOW.FileExistence	The existences of the file " <i>file</i> " must be checked with the instruction INQUIRE before being opened or created.
COM.FLOW.FilePath	It is not allowed to use directly the file name. Store the file path in a variable. Use the variable instead.
COM.FLOW.Recursion	The use of recursivity is not allowed.



Manuel Utilisateur i-Code CNES

Réf. : DCT/AQ /SO - 2014.0020026

Version : 1.2

Date : 16/11/2015

Page : 43/48

Règle	Message
COM.INST.BoolNegation	Double negation is not allowed.
COM.INST.Brace	Parentheses are needed for readability.
COM.INST.CodeComment	Commented code is not allowed. It shall be suppressed.
COM.INST.GOTO	The keyword GOTO is not allowed.
COM.INST.Line	More than one instruction per line is not allowed.
COM.INST.LoopCondition	A loop condition shall be written with inequality (.LE.,<=, or .GT.,>=)
COM.NAME.Homonymy	Names must be unique. The name "variable" is already defined in this file.
COM.PRES.Indent	The code is not indented.
COM.PRES.LengthLine	There are more than 100 characters in this line.
COM.PROJECT.Header	<ul style="list-style-type: none">- No file header existing. This module/function should have a header with a brief description.- No file header (file name not found). This module/function should have a header with a brief description.- The module/function should have a header with a brief description.
COM.TYPE.Expression	Mixed types " <i>type_variable_1</i> " with " <i>type_variable_2</i> " in the same expression

8.1.2. FORTRAN 77

Règle	Message
F77.BLOC.Common	Unnamed COMMON is not allowed.
F77.BLOC.Else	The IF instruction shall finish with an ELSE after the last ELSE IF.
F77.BLOC.Function	When calling a function, the brackets following the function name are mandatory.



Manuel Utilisateur i-Code CNES

Réf. : DCT/AQ /SO - 2014.0020026

Version : 1.2

Date : 16/11/2015

Page : 44/48

Règle	Message
F77.BLOC.Looping	Loops shall have distinct ends.
F77.DATA.Array	The dimension of the array "variable" is not well declared. The * shall be used for the last dimension.
F77.DATA.Common	The INCLUDE instruction shall be used to reference the needed common bloc.
F77.DATA.Double	The double precision variable is not correctly initialized. It misses the character D in its declaration.
F77.DATA.Initialization	The variable "variable" shall be initialized with DATA or BLOCK DATA before its use.
F77.DATA.IO	The use of * with logical units is not allowed.
F77.DATA.LoopingDo	The control variable in a loop shall be an integer.
F77.DATA.Parameter	"variable" belongs to parameter types forbidden when calling a function: a constant, an expression to be evaluated, a call to another function
F77.ERR.OpenRead	The status of OPEN/READ shall be tested with the parameter IOSTAT.
F77.INST.Assign	The instruction ASSIGN Is not allowed.
F77.INST.Dimension	The instruction DIMENSION Is not allowed.
F77.INST.Equivalence	The instruction EQUIVALENCE is not allowed.
F77.INST.Function	It misses the type declaration in FUNCTION header.
F77.INST.If	The arithmetic if is not allowed.
F77.INST.Include	The executable instruction "variable" is not allowed in the include file.
F77.INST.Pause	The instruction PAUSE is not allowed.
F77.INST.Return	The instruction RETURN(i) is not allowed.
F77.INST.Save	The instruction SAVE is only permitted for local variables



Manuel Utilisateur i-Code CNES

Réf. : DCT/AQ /SO - 2014.0020026

Version : 1.2

Date : 16/11/2015

Page : 45/48

Règle	Message
F77.MET.Line	There are more than 72 characters in this line.
F77.NAME.GenericIntrinsic	It should be used the generic name of the intrinsic function instead of "variable"
F77.NAME.Intrinsic	It is not allowed to use the name of an intrinsic function.
F77.NAME.KeyWords	The variable "variable" is a keyword in Fortran77 language.
F77.NAME.Label	The use of labels is not allowed except with the instructions FORMAT and CONTINUE.
F77.PROTO.Declaration	The function "variable" shall be declared.
F77.REF.IO	The logical entities shall be declared using a symbolic name.
F77.REF.Open	The instruction OPEN shall be called with the parameters FILE, STATUS and POSITION.
F77.REF.Parameter	It is not allowed to provide as a parameter the variables of an accessible bloc COMMON. The variable "variable" is used in a wrong way.
F77.TYPE.Basic	"variable" is not a basic type. Basic types are INTEGER, REAL, DOUBLE PRECISION, COMPLEX, LOGICAL and CHARACTER.
F77.TYPE.Hollerith	Type Hollerith is not allowed. "variable" shall be a CHARACTER.

8.1.3. FORTRAN 90

Règle	Message
F90.BLOC.File	The file "variable" is not correctly closed.
F90.DATA.Array	The array "variable" must not send the arguments inside the function.
F90.DATA.ArrayAccess	Array "variable1" initialized using other array named "variable2" with repeated values.
F90.DATA.ArrayDeclaration	The dimension variable "variable" of the array must be a



Manuel Utilisateur i-Code CNES

Réf. : DCT/AQ /SO - 2014.0020026

Version : 1.2

Date : 16/11/2015

Page : 46/48

n	parameter of a function.
F90.DATA.Constant	The constants shall be declared and initialized in a module.
F90.DATA.ConstantFloat	Float constant "variable" shall be declared using the subtype_parameter: <name>_<subtype_parameter>
F90.DATA.Float	It is not allowed to use the format * for reals like "variable".
F90.DATA.Parameter	It misses the use of intrinsic function SELECTED_REAL_KIND or SELECTED_INT_KIND for the subtype specification.
F90.DESIGN.Include	Is it possible to use a module instead of this inclusion?
F90.DESIGN.Interface	Interface Module shall only contain: INTERFACE, USE, IMPLICIT instructions as well as PRIVATE or PUBLIC declaration.
F90.DESIGN.IO	The value of the logic unity should be a integer or a variable initialised directly.
F90.DESIGN.Obsolete	<p>The instruction calculated GOTO is not allowed.</p> <p>The instruction PAUSE is not allowed.</p> <p>The alternate return statement is not allowed.</p> <p>There is a branch on an END IF statement. It is not allowed.</p> <p>The use of CHARACTER* is not allowed.</p> <p>The instruction HOLLERITH is not allowed inside FORMAT. Error in "variable" used.</p> <p>The instruction ASSIGN contains the label for the FORMAT instruction.</p> <p>Arithmetical IF is not allowed.</p> <p>A DO loop shall end with END DO.</p> <p>The variable "variable" is a real used in a do loop. Use only INTEGER.</p> <p>Each loop shall have its own END DO. Shared END DO is forbidden.</p>



Manuel Utilisateur i-Code CNES

Réf. : DCT/AQ /SO - 2014.0020026

Version : 1.2

Date : 16/11/2015

Page : 47/48

F90.ERR.Allocate	The status of the ALLOCATE or DEALLOCATE instruction is not checked
F90.ERR.OpenRead	<ul style="list-style-type: none">- There is no parameter IOSTAT in the OPEN/READ instruction.- The return of IOSTAT is no checked in the OPEN/READ instruction.
F90.INST.Associated	The pointer « variable » is not set to null before the use of the instruction ASSOCIATED.
F90.INST.Entry	The instruction ENTRY is not allowed.
F90.INST.Equivalence	The instruction EQUIVALENCE is not allowed.
F90.INST.If	Logical IF (without THEN and ENDIF) is only allowed with EXIT, CYCLE, GOTO, RETURN statements.
F90.INST.Intent	It misses the attribute INTENT for the parameter “variable”
F90.INST.Nullify	It misses the instruction NULLIFY after the DEALLOCATION of “variable”.
F90.INST.Only	The instruction ONLY must be preceded by a comment.
F90.INST.Operator	The symbolic notation (==, /=, <=, <, >=, >) must be used instead of (.EQ., .NE., .LT., .LE., .GT., .GE.). Error in “variable”.
F90.INST.Pointer	This use of POINTER is not allowed.
F90.NAME.GenericIntrinsic	Use the generic name of the intrinsic functions instead of “variable”.
F90.NAME.KeyWords	The variable “variable” is a keyword in Fortran90 language.
F90.PROTO.Overload	Overloading operator is not allowed. Overload of “variable”
F90.REF.ARRAY	It should be used the notation(:) to specify the entire use of the arrays: “list_variables”.
F90.REF.Interface	The function "function" is not visible in this point.
F90.REF.Label	It misses the name of the subprogram. It must finish with END TYPE_PROGRAM NAME.



Manuel Utilisateur i-Code CNES

Réf. : DCT/AQ /SO - 2014.0020026

Version : 1.2

Date : 16/11/2015

Page : 48/48

F90.REF.Open	It misses one or more parameters In OPEN instruction. Mandatory parameters are FILE, STATUS, IOSTAT, POSITION.
F90.REF.Variable	The variable "variable" is used with different names inside the subprogram.
F90.TYPE.Derivate	The variable " must be defined inside the module structure.
F90.TYPE.Integer	It misses the declaration SELECTED_INT_KIND in the initialisation of "variables"
F90.TYPE.Real	It misses the declaration SELECTED_REAL_KIND in the initialisation of "variables"