Table des matières

Branchement à l'aide du contrôle de flux 1
Branchement à l'aide du contrôle de flux : Introduction 1
Utilisation du contrôle de paires 1
Si / Fin si 2
Autrement si / Fin autrement si 3
Autrement / Fin autrement 5
Pendant / Fin pendant 6
Faire / Jusqu'à 8
Sélectionner / Fin Sélectionner 9
Cas / Fin Cas 11
Casse par défaut / Fin casse par défaut 12
Utilisation de boucles génériques 13
Création d'étiquettes 16
Saut à une étiquette avec Aller à 17
Branchement en cas d'erreur 19
Interfaces prises en charge en cas d'erreur 20
Branchement avec des sous-programmes 21
Création d'un sous-programme 22
Édition d'un sous-programme existant 26
Présentation de la boîte de dialogue Édition de l'argument 26
Appel d'un sous-programme 27
Présentation de la boîte de dialogue Appeler sous-programme 30
Exemples de sous-programmes 34
Exemple de code d'un sous-programme 37
Fin d'une routine de mesure 39

Branchement à l'aide du contrôle de flux

Branchement à l'aide du contrôle de flux : Introduction

Imaginez que votre pièce soit composée de nombreux éléments, mais que vous souhaitiez simplement en mesurer certains plusieurs fois afin d'obtenir un ensemble de données statistiques pour ces éléments . Imaginez également que vous souhaitiez passer à une pièce spécifique de votre routine de mesure selon les instructions de l'utilisateur. Vous pouvez effectuer ces tâches et bien d'autres à l'aide des commandes de contrôle de flux. Vous pouvez contrôler le flux de votre routine de mesure en définissant des conditions pour certaines commandes.

Ce chapitre vous fournit toutes les informations nécessaires pour ces tâches. Elle présente les instructions conditionnelles de syntaxe, les boucles et les sous-programmes. Vous y trouverez également de nombreux exemples de code.

Remarque : Lorsque le bouclage ou branchement apparaît dans les exemples de code, la mise en retrait sert à améliorer la lisibilité des instructions applicables à chaque condition. Dans le code affiché dans la fenêtre de modification, vous ne voyez apparaître aucune mise en retrait.

Les rubriques principales de ce chapitre sont les suivantes :

- Utilisation du contrôle de paires
- Utilisation de boucles génériques
- Création d'étiquettes
- Branchement en cas d'erreur
- Branchement avec des sous-programmes
- Fin d'une routine de mesure

Utilisation du contrôle de paires

Le sous-menu Insérer | Commande de contrôle de flux | Paires de contrôle propose diverses paires de commandes qui s'exécutent dans la fenêtre de modification afin de gouverner ou de "contrôler" le flux correct de la routine de mesure. Pour insérer une commande de type paire de contrôle dans la fenêtre de modification, il vous suffit de saisir la commande ou d'en choisir une dans ce sous-menu.

Important : Lorsque vous utilisez une instruction conditionnelle de branchement pour tester la valeur d'un commentaire OUI / NON, sachez que le test doit rechercher cette valeur « OUI » ou

« NON » en majuscules. Cette valeur en lettres minuscules ne fonctionne pas. Pour en savoir plus sur les commentaires, voir la rubrique « Insertion de commentaires de programmation », au chapitre « Insertion de commandes de rapport ».

Si / Fin si

L'option de menu Insérer | Commande de contrôle de flux | Paires de Contrôle | If / End If vous permet d'ajouter un bloc conditionnel à la routine de mesure. Les éléments entres les commandes IF et END IF ne sont exécutés que si l'expression de la commande IF est TRUE (non nulle). Dans le cas contraire, le flux de l'exécution passe à la première commande après la commande `END/IF`.

La ligne de commande de la fenêtre de modification pour une instruction IF / END IF est :

```
IF/expression
END_IF/
```

Pour insérer les commandes Si / Fin si :

1. Placez le curseur à l'emplacement souhaité dans la fenêtre Édition.
2. Sélectionnez **Si / Fin si** dans la barre de menus. L'instruction SI / FIN SI apparaît dans la fenêtre de modification.

Exemple de code pour Si / Fin si

Soit l'exemple suivant de code demandant à l'utilisateur s'il souhaite mesurer un élément de point.

```
C1= COMMENTAIRE/OUINON, Souhaitez-vous mesurer l'élément de point, PNT1 ?
IF/INPUTC1.==« OUI »
      PNT1 = ÉLÉM/POINT,RECT
      …
      …
      ENDMEAS/
END_IF/
```

Explication du code exemple

C1=COMMENTAIRE/OUINON
Cette ligne enregistre et stocke la réponse affirmative ou négative de l'utilisateur.

SI/INPUTC1.**==« OUI »**
Cette ligne constitue l'expression. Elle permet de vérifier que l'entrée de commentaire 1 est

OUI. S'il s'agit bien de OUI, l'instruction SI est VRAIE, les instructions après SI sont exécutées et, selon cet exemple, l'élément `PNT1` est mesuré. Si la réponse est NON, le programme passe directement à l'instruction FIN SI.

`FIN_SI`
Cette ligne termine l'exécution des commandes présentes dans le bloc de code SI / FIN SI. En d'autres termes, si l'utilisateur clique sur **Non** comme commentaire, PC-DMIS va à toute commande après cette ligne.

Autrement si / Fin autrement si

L'option de menu Insérer | Commande de contrôle de flux | Paires de Contrôle | Else If / End Else If vous permet d'ajouter un bloc conditionnel à la routine de mesure. Les éléments entre les commandes ELSE IF et END ELSE IF SI ne sont exécutés que si l'expression de la commande ELSE IF est vraie (non nulle). Le bloc ELSE IF / END ELSE IF doit être positionné juste *après* un bloc IF / END IF ou un autre bloc ELSE IF / END ELSE IF. Si toutes les expressions IF / ELSE IF au-dessus du bloc courant sont considérées comme fausses, alors l'expression est évaluée. Si l'expression est fausse (zéro), alors l'exécution passe à la commande suivante après la commande END ELSE IF. Si l'une des expressions IF / ELSE si au-dessus du bloc courant est vraie, tous les blocs ELSE IF / END ELSE IF après cette séquence sont ignorés.

La ligne de commande de la fenêtre de modification pour une instruction ELSE IF / END ELSE IF est :

```
ELSE_IF/expression
END_ELSE_IF/
```

Pour insérer les commandes AUTREMENT SI / FIN AUTREMENT SI :

1. Placez le curseur à l'emplacement désiré dans la fenêtre Édition, après une instruction existante SI/FIN SI ou une instruction AUTREMENT SI/FIN AUTREMENT SI.
2. Sélectionnez **Autrement si / Fin autrement si** dans la barre de menu. L'instruction AUTREMENT SI / FIN AUTREMENT SI apparaît dans la fenêtre de modification.

Remarque : Ce type de bloc est uniquement valide s'il est positionné *après* un bloc SI / FIN SI ou un bloc AUTREMENT SI / FIN AUTREMENT SI. Les paires de contrôle incorrectement positionnées sont affichées en rouge dans la fenêtre Édition.

Exemple de code pour Autrement si / Fin autrement si

Soit l'exemple suivant de code permettant d'afficher un message avertissant l'utilisateur lorsque l'une ou l'autre des valeurs X, Y, ou Z mesurée pour un point dépasse les tolérances définies.

```
PNT2 = ÉLÉM/POINT,RECT
…
```

```
…
FINMESURE/
SI/PNT2.X<6.9 OU PNT2.X>7.1
      COMMENTAIRE/OPER, « La valeur X mesurée pour PNT2 : » + PNT2.X + « est hors
      tolérance ».
FIN_SI/
AUTREMENT_SI/PNT2.Y<3.3 OU PNT2.Y>3.5
      COMMENTAIRE/OPER, « La mesure Y mesurée pour PNT2 : » + PNT2.Y + « est hors
      tolérance ».
FIN_AUTREMENT SI/
AUTREMENT_SI/PNT2.Z<.9 OU PNT2.Z>1.1
      COMMENTAIRE/OPER,« La valeur Z mesurée pour PNT2 : » + PNT2.Z + « est hors
      tolérance »
FIN_AUTREMENT SI/
```

Explication du code exemple

Ce code commence par vérifier la valeur X du point. Si la condition est fausse, le code vérifie la valeur Y. Si la condition pour la valeur Y est fausse, il vérifie la valeur Z.

Si l'une ou l'autre de ces conditions est vraie, PC-DMIS affiche le commentaire qui lui est associé et ignore les autres instructions conditionnelles de ce bloc de code.

SI/PNT2.X<6.9 OU PNT2.X>7.1
Cette ligne constitue l'expression. Elle vérifie que la valeur X mesurée n'est pas comprise entre 6,9 et 7,1. Si cette valeur est comprise entre ces limites, alors le premier commentaire est exécuté.

FIN_SI
Cette ligne termine l'exécution des commandes présentes dans le bloc de code SI / FIN SI. En d'autres termes, si la condition SI ALORS est fausse, PC-DMIS passe à toute commande après cette ligne.

AUTREMENT_SI/PNT2.Y<3.3 ou PNT 2.Y>3.5
Cette ligne constitue l'expression pour la première commande AUTREMENT_SI. Elle est uniquement exécutée si les conditions du bloc SI / FIN SI précédentes sont fausses. Elle vérifie que la valeur Y mesurée n'est pas comprise entre 3,3 et 3,5. Si cette valeur est comprise entre ces limites, le second commentaire est exécuté.

FIN_AUTREMENT SI/
Cette ligne termine l'exécution des commandes présentes dans le premier bloc de code AUTREMENT SI / FIN AUTREMENT SI.

AUTREMENT_SI/PNT2.Z<.9 OU PNT2.Z>1.1
Cette ligne est l'expression pour la seconde commande AUTREMENT SI. Elle est uniquement exécutée si les conditions du bloc ELSE IF / END ELSE IF précédentes sont fausses. Elle vérifie que la valeur Z est inférieure à 0,9 ou supérieure à 1,1. Si elle dépasse l'une de ces limites, elle exécute le troisième commentaire.

FIN_AUTREMENT SI/
Cette ligne termine l'exécution des commandes présentes dans le second bloc de code AUTREMENT SI / FIN AUTREMENT SI.

Autrement / Fin autrement

L'option de menu Insérer | Commande de contrôle de flux | Paires de Contrôle | Else / End Else vous permet d'ajouter un bloc conditionnel à la routine de mesure. Les éléments entre les commandes ELSE et END ELSE sont uniquement exécutés si tous les autres blocs if / end if et else if / end else if au-dessus du bloc else ont échoué (tous égaux à zéro). Les blocs ELSE / END ELSE doivent être situés à la fin d'une série de blocs IF / END IF ou ELSE IF / END ELSE IF afin d'être valides.

La ligne de commande de la fenêtre de modification pour une instruction ELSE / END ELSE est :

```
ELSE/
END_ELSE/
```

Pour insérer des commandes Autrement / Fin Autrement :

1. Placez le curseur à l'emplacement souhaité dans la *fenêtre Édition*. Notez que les blocs AUTREMENT / FIN AUTREMENT doivent être positionnés après une série de blocs SI / FIN SI ou AUTREMENT SI / FIN AUTREMENT SI.
2. Sélectionnez **Autrement / Fin autrement** dans la barre de menu. L'instruction AUTREMENT / FIN AUTREMENT apparaît dans la fenêtre de modification.

Exemple de code pour Autrement / fin autrement

Soit l'exemple suivant de code demandant à l'utilisateur s'il souhaite mesurer un élément de point.

```
C1= COMMENTAIRE/OUINON, Souhaitez-vous mesurer l'élément de point, PNT1 ? Cliquez sur NON
pour mesurer l'élément suivant.
IF/INPUTC1.==« OUI »
      PNT1 = ÉLÉM/POINT,RECT
      …
      …
      ENDMEAS/
END_IF/
AUTREMENT
      PNT2 = ÉLÉM/POINT, RECT
      …
      …
      ENDMEAS/
END ELSE
```

Explication du code exemple

C1=COMMENTAIRE/OUINON
Cette ligne enregistre et stocke la réponse affirmative ou négative de l'utilisateur.

SI/INPUTC1.**==« OUI »**
Cette ligne constitue l'expression. Elle permet de vérifier que l'entrée de commentaire 1 est OUI. S'il s'agit bien de OUI, l'instruction SI est VRAIE, les instructions après SI sont exécutées et, selon cet exemple, l'élément PNT1 est mesuré. Si la réponse est NON, le programme passe directement à l'instruction FIN SI.

END_IF
Cette ligne termine l'exécution des commandes présentes dans le bloc de code IF / END IF. En d'autres termes, si l'utilisateur clique sur **Non** comme commentaire, PC-DMIS va à toute commande après cette ligne.

ELSE
Si les conditions du bloc IF / END IF précédent sont fausses, les lignes de commande après cette ligne et qui précèdent celle END_ELSE sont exécutées. Dans ce cas, PNT2 est alors exécuté.

END_ELSE
Cette ligne termine l'exécution des commandes présentes dans le second bloc de code ELSE / END_ELSE.

Pendant / Fin pendant

L'option de menu Insérer | Commande de contrôle de flux | Paires de Contrôle | While / End While vous permet d'ajouter une boucle conditionnelle à la routine de mesure. Les éléments entre les commandes WHILE et END WHILE sont exécutés dans une boucle jusqu'à ce que la condition (ou expression) conservant la boucle active ne soit plus satisfaite, à savoir que l'expression de la boucle WHILE soit FALSE (nulle). La commande WHILE peut être ajoutée n'importe où dans la routine de mesure. L'expression est testée au début de chaque boucle.

La ligne de commande de la fenêtre de modification pour une instruction WHILE / END WHILE est :

```
WHILE/expression
END_WHILE/
```

Pour insérer une commande PENDANT / FIN PENDANT :

1. Placez le curseur à l'emplacement souhaité dans la fenêtre Édition.
2. Sélectionnez **Pendant / fin pendant** dans la barre de menus. L'instruction PENDANT / FIN PENDANT apparaît dans la fenêtre de modification.

Exemple de code pour Pendant / Fin pendant

Soit l'exemple suivant de code permettant à l'utilisateur de spécifier pour un élément le nombre de mesures effectuées par la routine de mesure.

```
C1=COMMENTAIRE/ENTRÉE, Combien de fois souhaitez-vous mesurer PNT1 ? Veuillez taper un
entier uniquement.
ASSIGN/COMPTE = 0
PENDANT/COMPTE < C1.INPUT
      PNT2 = ÉLÉM/POINT,RECT
      …
      …
      …
      FINMESURE/
      ASSIGN/COMPTE = COMPTE + 1
      COMMENTAIRE/OPER,« Mesuré » + COMPTE + « hors de » + INPUT C1.+ « fois ».
FIN_PENDANT/
```

Explication du code exemple

C1=COMMENTAIRE/ENTRÉE
Cette ligner permet d'enregistrer et de stocker l'entier saisi par l'utilisateur dans la variable `C1.INPUT`.

ASSIGN/COMPTE = 0
Cette ligne initialise `COMPTE`, une variable définie par l'utilisateur, et lui attribue la valeur 0. Le code utilise cette variable pour compter le nombre de fois que PC-DMIS mesure l'élément dans la boucle.

PENDANT/COMPTE < C1.INPUT
Cette ligne constitue l'expression. Elle permet de vérifier si la valeur `COMPTE` (initialement égale à 0) est inférieure à l'entier spécifié par l'utilisateur. Si la condition est vraie, les instructions présentes dans le bloc de code entre `PENDANT/` et `FIN_PENDANT/` sont exécutées.

ASSIGN/COMPTE = COMPTE + 1
Cette ligne permet d'incrémenter la variable `COMPTE` de 1, de façon à sortir de la boucle lorsque la condition devient fausse.

COMMENTAIRE/OPER,« Mesuré » + COMPTE + « hors de » + INPUT C1.+ « fois ».
Cette ligne affiche un message indiquant le nombre de mesures, par rapport au total, que la boucle permet d'exécuter.

END_WHILE
Cette ligne termine l'exécution des commandes présentes dans le bloc de code WHILE / END WHILE tant que la condition est fausse. Sinon, lorsque PC-DMIS rencontre cette commande, sa boucle revient à l'instruction `PENDANT`.

Faire / Jusqu'à

L'option de menu Insérer | Commande de contrôle de flux | Paires de Contrôle | Do / Until vous permet d'ajouter une boucle conditionnelle à la routine de mesure. Les éléments entre les commandes DO et UNTIL sont exécutés dans une boucle jusqu'à ce que l'expression de la commande UNTIL soit TRUE (non nulle). Les commandes DO/ UNTIL peuvent être ajoutées à tout emplacement de la routine de mesure. L'expression est testée à la fin de chaque boucle.

La ligne de commande de la fenêtre de modification pour une instruction DO / UNTIL est :

```
DO/
UNTIL/ expression
```

Pour insérer des commandes FAIRE / JUSQU'À :

1. Placez le curseur à l'emplacement souhaité dans la fenêtre Édition.
2. Sélectionnez **Do / Until** dans la barre de menus. Les instructions FAIRE / JUSQU'À apparaissent dans la fenêtre de modification.

Exemple de code pour Faire / Jusqu'à

Soit l'exemple suivant de code permettant à l'utilisateur de spécifier pour un élément le nombre de mesures effectuées par la routine de mesure. Cet exemple est similaire à celui dans la rubrique While / End While, sauf que PC-DMIS vérifie la condition à la fin de la boucle et non au début de celle-ci.

```
C1 = COMMENTAIRE/ENTRÉE,Saisissez le nombre de mesures que PC-DMIS doit effectuer sur
l'élément PNT1 : (veuillez taper uniquement un entier)
ASSIGN/COMPTE = 0
FAIRE/
      PNT1 = ÉLÉM/POINT,RECT
      …
      …
      FINMESURE/
      ASSIGN/COMPTE = COMPTE + 1
      COMMENTAIRE/OPER,« Mesuré » + COMPTE + « hors de » + INPUT C1.+ « fois ».
JUSQU'À/COMPTE == INPUT C1.
```

Explication du code exemple

C1=COMMENTAIRE/ENTRÉE
Cette ligner permet d'enregistrer et de stocker l'entier saisi par l'utilisateur dans la variable `C1.INPUT`.

ASSIGN/COMPTE = 0
Cette ligne initialise COMPTE, une variable définie par l'utilisateur, et lui attribue la valeur 0. Le code utilise cette variable pour compter le nombre de fois que PC-DMIS mesure l'élément dans la boucle.

DO/
Permet de débuter la boucle DO / UNTIL. Toutes les instructions sont exécutées au moins une fois et le flux de la routine de mesure sort de la boucle lorsque l'expression est fausse.

ASSIGN/COMPTE = COMPTE + 1
Cette ligne permet d'incrémenter la variable COMPTE de 1, de façon à sortir de la boucle lorsque la condition devient fausse.

COMMENTAIRE/OPER,« Mesuré » + COMPTE + « hors de » + INPUT C1.+ « fois ».
Cette ligne affiche un message indiquant le nombre de mesures, par rapport au total, que la boucle permet d'exécuter.

JUSQU'À/COMPTE == C1.INPUT
Cette ligne termine l'exécution des commandes à l'intérieur de la boucle FAIRE / JUSQU'À tant que la condition est fausse. Sinon, lorsque PC-DMIS rencontre cette commande, la boucle revient à l'instruction FAIRE.

Sélectionner / Fin Sélectionner

L'option de menu **Insérer | Commande de contrôle de flux | Paires de contrôle | Sélectionner / Fin Sélectionner** permet d'ajouter un bloc conditionnel qui est utilisé en conjonction avec les paires CASSE / FIN CASSE et Casse par défaut / Fin casse par défaut. L'expression de la commande SÉLECTION fournit les données qui sont comparées à l'expression dans les instructions de Casse. Si les deux expressions sont équivalentes, alors les instructions dans le bloc Casse / Fin Casse sont exécutées. Le bloc SÉLECTION / FIN SÉLECTION entoure les séries de blocs CASSE / FIN CASSE et CASSE PAR DÉFAUT / FIN CASSE PAR DÉFAUT.

La ligne de commande de la fenêtre de modification pour une instruction SELECT / END SELECT est :

```
SELECT/expression
END_SELECT/
```

Pour insérer les commandes Sélect / Fin sélect :

1. Placez le curseur à l'emplacement souhaité dans la fenêtre Édition.
2. Choisissez **Sélection / Fin Sélection** dans la barre de menus. Les instructions SÉLECTION / FIN SÉLECTION apparaissent dans la fenêtre de modification.

Exemple de code pour Sélectionner / Fin sélectionner

Les paires `SÉLECTION` / `FIN_SÉLECTION`, `CASSE` / `FIN_CASSE`, `CASSE PAS DÉFAUT` / `FIN_CASSE PAR DÉFAUT` fonctionnent toutes ensemble pour évaluer plusieurs conditions en offrant toute une gamme de possibilités.

Imaginez disposer de cinq cercles étiquetés CER1 à CER5 et que l'opérateur doive en mesurer un en appuyant simplement sur une touche du clavier. Vous pouvez alors utiliser un code comme celui-ci :

Code complet

```
FAIRE/
      C1=COMMENTAIRE/ENTRÉE, Saisissez un nombre pour la mesure de ce cercle :
      , POUR CER1 - Tapez 1
      , POUR CER2 - Tapez 2
      , POUR CER3 - Tapez 3
      , POUR CER4 - Tapez 4
      , POUR CER5 - Tapez 5
      , Tout autre caractère entraîne la sortie de la boucle
      SÉLECTION/C1.INPUT
            CASSE/1
                  CER1=ÉLÉMT/CERCLE
                  …
                  …
                  FINMESURE/
            FIN_CASSE
            CASSE/2
                  CER2 = ÉLÉMT/CERCLE
                  …
                  …
                  FINMESURE/
            FIN_CASSE
            CASSE/3
                  CER3 = ÉLÉMT/CERCLE
                  …
                  …
                  FINMESURE/
            FIN_CASSE
            CASSE/4
                  CER4 = ÉLÉMT/CERCLE
                  …
                  …
                  FINMESURE/
            FIN_CASSE
```

```
            CASSE/5
                  CER5 = ÉLÉMENT/CERCLE
                  …
                  …
                  FINMESURE/
            FIN_CASSE
            CASSE PAR DÉFAUT
                  COMMENTAIRE/OPER, sortie de la boucle maintenant.
            FIN_CASSE PAR DÉFAUT
      FIN_SÉLECTION
JUSQU'À C1.INPUT < 1 OU C1.INPUT > 5
```

Explication du code exemple

`SÉLECTION/C1.INPUT`
Cette ligne de code permet d'enregistrer une valeur numérique ou alphanumérique (dans ce cas, un nombre) saisie par l'utilisateur et détermine quel bloc `CASSE/FIN_CASSE` s'exécutera à partir de l'entrée. Remarquez que la paire `SÉLECTION / FIN_SÉLECTION` entoure le bloc de code complet. Toutes les paires `CASE / END_CASE` et `DEFAULT CASE / END_DEFAULT_CASE` doivent être incluses entre ces deux lignes.

`FIN_SÉLECTION`
Cette instruction marque la fin du code contenu entre les instructions de la paire SÉLECTION / FIN SÉLECTION.

`CASSE/1` à **`CASSE/5`**
En fonction de la valeur de `C1.INPUT`, l'un des blocs de code de type CASSE s'exécute. Par exemple, si `C1.INPUT` est égale à 1, le bloc CASSE 1 s'exécute pour la mesure de CIR1. Si la variable est égale à 2, alors le bloc de code CASSE 2 s'exécute pour la mesure de CER2, et ainsi de suite.

`FIN_CASSE`
Ces lignes terminent les blocs de code casse spécifiques.

`CASSE PAR DÉFAUT`
Si la valeur de `C1.INPUT` ne correspond à aucune instruction définie pour CASSE (si la valeur n'est pas un nombre de un à cinq), le bloc de code `CASSE PAR DÉFAUT` s'exécute. Dans ce cas, un message s'affiche et vous informe que vous quittez la boucle.

Notez que la boucle FAIRE / JUSQU'À entoure le bloc de code complet de l'exemple. Cela permet à l'utilisateur de continuer de choisir dans le menu créé à partir de la ligne `COMMENTAIRE/ENTRÉE` jusqu'à ce qu'il sélectionne un caractère qui n'est pas reconnu par les instructions CASSE.

Cas / Fin Cas

L'option de menu Insérer | Commande de contrôle de flux | Paires de Contrôle | Case / End Case vous permet d'ajouter un bloc conditionnel à la routine de mesure. Les éléments entre les commandes CASE et END CASE sont exécutés si l'expression de l'instruction CASSE donne

une valeur égale à l'expression de la commande SELECT correspondante. Dans le cas contraire, le bloc d'instructions est ignoré. Le bloc d'instructions CASE / END CASE doit être placé directement après une commande SELECT ou END CASE d'un bloc CASE / END CASE précédent. En outre, PC-DMIS ne peut pas comparer plusieurs expressions sur une seule instruction de casse.

La ligne de commande de la fenêtre de modification pour une instruction CASE / END CASE est :

```
CASE/expression
END_CASE/
```

Pour insérer la commande CASSE / FIN CASSE :

1. Placez le curseur à l'emplacement souhaité dans la fenêtre Édition. Notez les conditions de position mentionnées ci-dessus.
2. Sélectionnez **Case / End Case** dans la barre de menus. Les instructions CASE / END CASE s'affichent dans la fenêtre de modification.

Casse par défaut / Fin casse par défaut

L'option de menu Insérer | Commande de contrôle de flux | Paires de Contrôle | Default Case / End Default Case vous permet d'ajouter un bloc conditionnel à la routine de mesure. Les éléments entre les commandes DEFAULT CASE et END DEFAULT CASE sont exécutés si toutes les autres expressions des blocs CASSE / FIN CASSE précédents au sein du bloc CASE / END CASE correspondant sont fausses. Un seul bloc DEFAULT CASE / END DEFAULT CASE est autorisé au sein d'un bloc SELECT/ END SELECT. Le bloc DEFAULT CASE / END DEFAULT CASE doit être placé après tous les blocs CASE / END CASE au sein du bloc SELECT / END SELECT.

La ligne de commande de la fenêtre de modification pour une instruction DEFAULT CASE / END DEFAULT CASE est :

```
DEFAULT CASE/
END_DEFAULT_CASE/
```

Pour insérer les commandes CASSE PAR DÉFAUT/ FIN CASSE PAR DÉFAUT :

1. Placez le curseur à l'emplacement souhaité dans la fenêtre Édition en notant les conditions de position mentionnées ci-dessus.
2. Sélectionnez **Default Case / End Default Case** dans la barre de menus. Les instructions DEFAULT CASE / END DEFAULT CASE s'affichent dans la fenêtre de modification.

Utilisation de boucles génériques

Boîte de dialogue Paramètres de boucle

L'option **Insérer | Commande de contrôle de flux | Boucle** ouvre la boîte de dialogue **Paramètres de boucle**. Vous pouvez utiliser cette boîte de dialogue pour créer une commande LOOP répétant la routine de mesure (ou des parties de la routine de mesure) avec ou sans les décalages. La commande BOUCLE peut être ajoutée à tout emplacement de la routine de mesure, bien que cette fonction soit plus utile en début et en fin de programme.

N° de début - Cette zone spécifie à PC-DMIS le numéro de la position de début dans une série de pièces.

> **Exemple :** Vous avez 10 pièces et vous souhaitez débuter à la position numéro 5 : Entrez 10 comme nombre total de pièces et 5 comme position de début.

N° de fin - Cette zone indique à PC-DMIS combien de fois faire une boucle dans la routine de mesure. Ce nombre est habituellement le même que le nombre de pièces contenues dans le montage (ou le nombre de modèles sur la pièce) dans la direction x (y ou z). PC-DMIS requiert aussi le numéro de la pièce (modèle) de début.

> **Exemple :** vous avez 10 pièces dans la direction x (y ou z) et vous voulez commencer à la position numéro 5. Entrez 10 dans la zone **N° de fin**. Entrez 5 dans la zone **N° de début**.

Sauter numéro - Dans une boucle, PC-DMIS répète une routine de mesure le nombre de fois indiqué. La zone **Sauter numéro** vous permet de sauter un incrément spécifié. Vous ne pouvez pas *sauter* le *premier* incrément utilisant cette zone, mais vous pouvez changer le **numéro de début** à 2 et effectivement le sauter de cette façon.

> **Exemple :** vous pouvez définir le paramètre pour ignorer chaque fois le troisième incrément de la boucle. Si le nombre 3 est spécifié, PC-DMIS mesure les première et deuxième pièces et passe à la quatrième pièce.

Décalage - La zone de décalage contient trois cases :

Axe X, **Axe Y**, **Axe Z** - Ces zones indiquent le décalage x, y ou z entre des pièces ou des modèles sur la même pièce. Elles décalent la pièce de la distance entrée chaque fois que la boucle est effectuée. Le premier décalage est basé sur l'origine de la pièce.

Angle - Cette zone définit le décalage angulaire entre les pièces ou les modèles sur la même pièce. Le premier décalage est basé sur l'origine de la pièce. PC-DMIS décale la pièce de la valeur d'angle chaque fois que la boucle est effectuée.

Si une commande d'alignement figure dans une boucle et que celle-ci utilise des décalage, vous *devez* définir tous les axes pour cet alignement. Par ailleurs, l'alignement dans la boucle doit utiliser des éléments mesurés dans la boucle.

ID de boucle - Cette case à cocher indique si PC-DMIS incrément l'ID d'élément (dans la loupe) comme valeur entre crochets correspondant à l'incrément de boucle.

Exemple : CER1 devient CER1[1] sur la première boucle, CER1[2] sur la deuxième boucle, et ainsi de suite.

ID d'éléments dans des bases de données statistiques

Si vous cochez la case **ID de boucle** et envoyez des données statistiques à une base de données, PC-DMIS n'affiche parfois pas ces ID dans la base de données.

Examinez la procédure suivante :

Avec une commande STATS/ONet STATS/UPDATE*dans une boucle de routine*, les ID de boucle n'apparaissent pas dans la base de données.

Avec une commande STATS/ON *hors d'une boucle de routine* et une commande STATS/UPDATE *dans une boucle de routine*, les ID de boucle n'apparaissent pas dans la base de données.

Avec une commande STATS/ON *hors d'une boucle de routine* et une commande STATS/UPDATE *hors d'une boucle de routine*, les ID de boucle apparaissent dans la base de données.

Avec une commande STATS/ON *dans une boucle de routine* et une commande STATS/UPDATE *hors d'une boucle de routine*, les ID de boucle apparaissent dans la base de données.

ID de variable - Cette zone vous permet de définir le nom de la variable utilisé pour suivre l'itération en cours de la boucle (ou la boucle en cours parmi le nombre de boucles spécifiées). Pendant l'exécution de la routine de mesure, cette variable est égale au nombre actuel d'itérations de la boucle.

Fin de boucle - Ce bouton termine le processus de boucle. La commande `LOOP/START` doit être suivie par la commande `LOOP/END` dans la fenêtre de modification.

Utilisation de la fonction de boucle

L'option de boucle a trois utilisations principales :

- Vous avez un montage de plusieurs pièces qui maintient une grille de pièces. Le montage doit utiliser un espacement homogène entre les rangées. Les décalages de translation / rotation permettent d'indexer d'une pièce à une autre dans la grille de pièces.
- Vous avez un montage qui maintient une pièce et vous souhaitez basculer vers une autre pièce avant chaque boucle de la routine. Une commande COMMENT est utile pour arrêter la MMT lorsque la pièce est remplacée par une nouvelle. La commande peut se trouver au début ou à la fin de la boucle.
- Vous souhaitez utiliser l'option BOUCLE pour faire pivoter la routine de mesure afin de mesurer une portion différente de la même pièce. Par exemple, vous pouvez créer une routine de mesure pour mesurer un modèle d'alésage complexe dupliqué 10 fois sur la pièce. La routine de mesure n'a alors à mesurer qu'un seul alésage. L'option LOOP peut être utilisée pour décaler cette routine de mesure et mesurer les 9 autres occurrences du modèle.

Remarque : Si vous utilisez un alignement dans une boucle, PC-DMIS vous permet d'employer l'alignement actif dans la ligne de commande `ALIGNEMENT/START` au lieu de toujours rappeler un alignement stocké auparavant. Voir la rubrique "Utilisation d'un alignement dans des boucles" au chapitre "Création et utilisation d'alignements".

Création d'une boucle

1. Sélectionner **Insérer | Commande de contrôle de flux | Boucle** dans la barre de menus pour ouvrir la boîte de dialogue **Paramètres de la fonction de boucle**.
2. Modifiez les cases selon vos besoins.
3. Sélectionnez les paramètres requis (comme Nombre de pièces, N° de début, Ignorer numéro, Angle de décalage).
4. Dans la fenêtre Édition, placez le curseur où vous voulez commencer la boucle.
5. Cliquez sur le bouton **OK**.

La ligne de commande de la fenêtre Édition pour la boucle a l'aspect suivant :

```
NOMVAR = BOUCLE/DÉBUT, ID = O/N, NUMÉRO = 0, DÉBUT = 1, IGNORER = ,
DÉCALAGE: AXEX = 0, AXEY = 0, AXEZ = 0, ANGLE = 0
```

Remarque : pour terminer la procédure de création de boucle, vous devez finir avec une commande `LOOP/END`. PC-DMIS met en boucle toutes les commandes de la fenêtre de modification incluses dans les commandes `LOOP/START` et `LOOP/END`. Vous pouvez insérer cette commandes de l'une de ces façons : dans la fenêtre de modification, entrez `LOOP/END`, sélectionnez l'option **Insérer | Command de contrôle de flux | Fin de boucle** ou dans la boîte de dialogue **Paramètres de boucle**, cliquez sur le bouton **Fin de boucle**.

Création d'étiquettes

Vous pouvez utiliser une commande `LABEL` avec une commande `GOTO` ou `IF_GOTO` pour contrôler à quel endroit le flux d'exécution est envoyé selon certaines conditions. Le nom de l'étiquette ne peut pas comporter d'espaces et ne doit pas dépasser 230 caractères. PC-DMIS affiche le nom de l'étiquette en majuscules.

Pour créer une commande `LABEL`, procédez de l'une des façons suivantes :

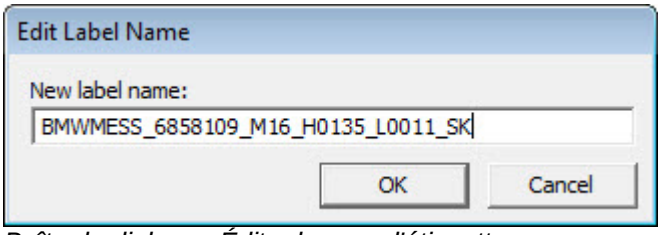
Entrez la commande.

1. Placez le curseur sur une ligne vide dans la fenêtre de modification.
2. Entrez LABEL et appuyez sur la touche Tab.
3. Dans la zone mise en évidence, entrez un nouveau nom d'étiquette.

ou

Choisissez la commande.

1. Placez le curseur dans la fenêtre de modification.
2. Dans la barre de menus, sélectionnez **Insérer | Commande de contrôle de flux | Étiquette** pour accéder à la boîte de dialogue **Modifier le nom d'étiquette**.
3. Dans la zone **Nouveau nom d'étiquette**, entrez le nom de l'étiquette.



Boîte de dialogue Éditer le nom d'étiquette

4. Cliquez sur **OK** pour insérer le nom de l'étiquette à l'emplacement suivant dans la fenêtre de modification.

La commande LABEL dans la fenêtre de modification en mode commande est :
`ID = LABEL/`

Où ID correspond au nom d'étiquette indiqué.

Pour des informations sur l'utilisation de la commande `GOTO` avec des étiquettes, voir la rubrique "Saut à une étiquette avec GOTO".

Pour des informations sur l'utilisation de la commande `IF_GOTO` avec des étiquettes, voir la rubrique "Saut à une étiquette selon certaines conditions".

Saut à une étiquette avec Aller à

Vous pouvez commander à votre routine de mesure de sauter à une étiquette spécifique en créant une commande GOTO.

Pour créer une commande GOTO, procédez de l'une des façons suivantes :

Entrez la commande.

1. Placez le curseur sur une ligne vide dans la fenêtre de modification.
2. Entrez GOTO et appuyez sur la touche Tab.
3. Entrez le nom de l'étiquette à laquelle sauter.

ou

Choisissez la commande.

1. Placez le curseur dans la fenêtre de modification.
2. Dans la barre de menus, sélectionnez **Insérer | Commande de contrôle de flux | Aller à** pour accéder à la boîte de dialogue **Aller à**.

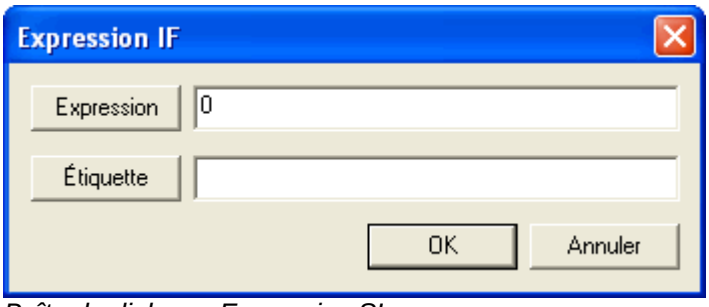
Boîte de dialogue Aller à

3. Si des étiquettes existent déjà dans la routine de mesure, elles apparaissent dans la zone **Étiquettes en cours**.
4. Dans la zone **Étiquettes en cours**, sélectionnez l'étiquette souhaitée ou, dans la zone **Aller à l'étiquette**, entrez directement un nom.
5. Cliquez sur **OK** pour insérer la commande ALLER À à l'emplacement suivant possible dans la fenêtre de modification.

La commande GOTO dans la fenêtre de modification en mode commande est :
`GOTO/ID_étiquette`

Où ID_étiquette correspond au nom de l'étiquette indiquée. Le nom de l'étiquette ne peut pas comporter d'espaces et ne doit pas dépasser 230 caractères.

Remarque : Si l'étiquette n'a pas déjà été créée, son nom apparaît en évidence en rouge dans la fenêtre de modification en mode commande et la commande ALLER À est ignorée lors de l'exécution.



Boîte de dialogue Expression SI

Avec l'option **Insérer | Commande de contrôle de flux | If Goto**, vous pouvez utiliser la boîte de dialogue **Expression If** pour créer des instructions IF GOTO dans votre routine de mesure. Quand la routine de mesure est exécutée et que PC-DMIS rencontre une instruction IF GOTO, le flux de la routine va à l'identification de l'étiquette si l'expression indiquée est évaluée à une valeur non nulle.

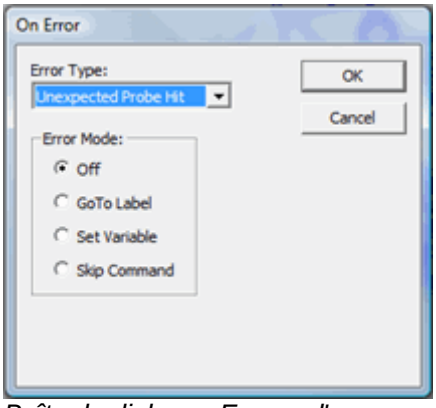
La ligne de commande de la fenêtre de modification pour une instruction IF_GOTO est :

```
IF_GOTO/expression, GOTO=Label
```

Expression - Si vous cliquez sur **Expression**, le générateur d'expressions s'ouvre. Ce dernier permet de créer une variété d'expressions pouvant être utiles dans votre routine de mesure. Une fois créée, l'expression apparaît dans la zone **Expression**. Pour des informations sur la création d'expressions, voir le chapitre "Utilisation d'expressions et de variables".

Étiquette - Si vous cliquez sur **Étiquette**, la boîte de dialogue **Goto** s'ouvre. Dans la boîte de dialogue **Goto**, vous pouvez choisir l'étiquette auquel le flux de routine doit aller quand l'expression définie est rencontrée. L'étiquette choisie apparaît dans la zone **Étiquette**. Si vous connaissez le nom de l'étiquette, vous pouvez aussi l'entrer dans la zone. Pour en savoir plus sur les étiquettes, voir "Saut à une étiquette".

Branchement en cas d'erreur



Boîte de dialogue En cas d'erreur

La commande **Insérer | Commande de contrôle de flux | En cas d'erreur** ouvre la boîte de dialogue **En cas d'erreur**. Vous pouvez utiliser cette boîte de dialogue pour commander à PC-DMIS l'action à exécuter quand une erreur de la machine se produit.

Type d'erreur - PC-DMIS fait le suivi de ces conditions d'erreur :

- Palpage imprévu
- Palpage manqué
- Réflecteur introuvable (utilisé avec le suiveur de tâche dans PC-DMIS Portable)
- Erreur Laser (voir « Gestion des erreurs du palpeur Laser avec EN CAS D'ERREUR », si vous utilisez une configuration Laser)

Mode erreur - Pour chacune de ces conditions d'erreur, les actions suivantes sont possibles.

- **Étiquette GoTo** - Le flux de la routine de mesure va à une étiquette définie. Voir "Utilisation d'étiquettes".
- **Définir variable** - Définit la valeur d'une variable à un.
- **Désactivé** - PC-DMIS ne fait rien.
- **Ignorer commande** - La routine de mesure ignore la commande actuelle et passe à la commande suivante marquée dans la routine de mesure.

Par défaut, toutes les routines de mesure commencent par l'action pour les deux types d'erreurs définis à **Désactivé** (ne rien faire). Le mode d'action pour chaque type d'erreur peut être changé au cours de la routine de mesure.

ID étiquette - Vous permet d'entrée une référence à une étiquette qui n'existe pas encore.

Étiquettes courantes - Répertorie toutes les étiquettes dans la routine de mesure.

Exemple : si, durant l'éxecution, PC-DMIS rencontre une commande `EN CAS D'ERREUR/ Palpage imprévu / Sauter à étiquette`, tout palpage imprévu survenant après ce point dans la routine de mesure entraîne l'exécution d'un saut à l'étiquette spécifiée. L'action 'Définir

une valeur de variable à un' entraîne la définition de la variable dès que le type d'erreur spécifié se produit. Cette valeur de la variable peut ensuite être testée à l'aide d'une instruction SI afin de provoquer l'exécution d'un saut à un nouveau point dans la routine de mesure.

Pour utiliser la commande En cas d'erreur :

1. Sélectionnez **Insérer | Commande de contrôle de flux | En cas d'erreur** pour ouvrir la boîte de dialogeu **En cas d'erreur**.
2. Dans la liste **Type d'erreur**, sélectionnez le type d'erreur comme condition pour le branchement.
3. Dans la zone **Mode erreur**, sélectionnez l'une des options pour définir l'action devant s'exécuter quand cette condition est remplie.
4. Cliquez sur le bouton **OK** pour ajouter la commande `ONERROR`. Le bouton **Annuler** ferme la boîte de dialogue **En cas d'erreur** sans appliquer les changements.

La ligne de commande de la fenêtre de modification pour cette commande a l'aspect suivant :

```
ONERROR/<type>,<mode>
```

Où <type> correspond au type d'erreur et <mode> au mode d'erreur à prendre.

Interfaces prises en charge en cas d'erreur

Toutes les interfaces ne prennent pas en charge la commande En cas d'erreur. Consultez le tableau suivant pour voir si votre interface est prise en charge.

- Si votre interface apparaît dans la liste, une petite case noire indique quel est le type d'erreur pris en charge par cette interface.
- Si votre interface n'apparaît pas dans la liste, la commande En cas d'erreur ne peut pas être utilisée.

Interfaces prises en charge	Palpage imprévu	Palpage manqué	Réflecteur introuvable
Standard B&S	■	■	-
DEA		■	-
Élm	■	■	-
Fédéral/Renault	■	■	-
Johansson	■	■	-
Pisteur Leica			■
Leitz	■	■	-
LK Direct (également connue sous le nom de LKRS232)	■	■	-
Pilote LK	■	■	-
Metrolog	■	■	-
Mitutoyo Bright	■	■	-

Mitutoyo	■	■	-
Mora	■	■	-
Omnitech	■	■	-
Renishaw	■	■	-
Sharpe	■	■	-
Sheffield	■	■	-
Wenzel	■	■	-
Zeiss	■	■	-

Branchement avec des sous-programmes

Les sous programmes sont des blocs de codes dans votre routine de mesure ou dans une routine de mesure externe qui sont habituellement référencées de façon répétée, permettant ainsi une programmation plus concise. PC-DMIS vous permet de transmettre des informations à des "arguments" (ou variables locales) dans le sous-programme. Les types d'arguments qui peuvent être transmis à un sous-programme sont des valeurs numériques, des variables, des chaînes de texte et des noms d'éléments.

Indication : Les blocs de commande de sous-programme sont inclus dans les commandes SOUS-PROGRAMME et FINSP.

Une fois que vous avez créé un sous-programme dans votre routine de mesure, vous pouvez l'appeler à partir de votre routine de mesure courante ou d'une autre routine de mesure, ce qui provoque l'envoi du flux d'exécution de la routine de mesure dans le sous-programme spécifié, exécutant des commandes contenues dans le bloc de commande du sous-programme. Puis, le flux de la routine de mesure retourne à l'instruction suivant immédiatement l'instruction d'appel.

Indication : Vous appelez les sous-programmes en utilisant la commande APPELSP.

Sous-programmes externes

Les sous-programmes externes ou situés dans une routine de mesure hors de la routine de mesure exécutante, n'ont pas accès aux éléments, variables, ou alignements à partir de la routine de mesure exécutante. Le sous-programme a toujours accès aux éléments dans sa propre routine de mesure. La routine de mesure externe et la routine de mesure exécutante doivent utiliser les mêmes unités de mesure.

Imbrication de sous-programmes

Vous pouvez imbriquer des sous-programmes dans d'autres sous-programmes. La seule limite au nombre d'arguments et aux sous-programmes imbriqués est la quantité de mémoire disponible.

Création d'un sous-programme

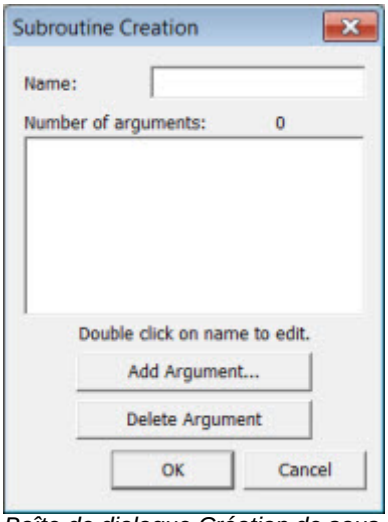
Création d'un sous-programme en tapant SOUS-PROGRAMME

Vous pouvez insérer cette commande en entrant simplement SOUS-PROGRAMME dans le mode de commande de la fenêtre d'édition et en appuyant ensuite sur TAB. Une fois la commande insérée, vous devez spécifier le nom du sous-programme et tous ses arguments. Consultez la syntaxe du sous-programme et l'exemple ci-dessous pour obtenir cette information.

Tapez la commande FINSP et appuyez sur TAB pour terminer le bloc de commande. Toute commande de la fenêtre Édition placée dans ce bloc de commande sera considérée comme faisant partie du sous-programme et exécutée quand le sous-programme est appelé.

Création d'un sous-programme en utilisant l'élément de menu Sous-programme

1. Sélectionnez **Insérez | Commande de contrôle de flux | Sous-programme** dans le sous-menu. La boîte de dialogue **Création de sous-programme** s'ouvre. Pour des informations sur cette boîte de dialogue, voir la rubrique « Présentation de la boîte de dialogue Création de sous-programme ».



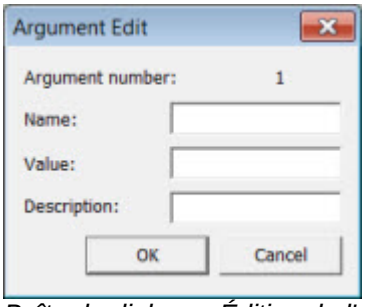
Boîte de dialogue Création de sous-programme

2. Donnez un nom au sous-programme en le tapant dans la case **Nom**. Le nom est limité à 180 caractères ou moins. Si vous entrez 181 caractères, le message d'erreur « Ligne trop longue » apparaît.

   Quand vous cliquez sur le bouton **OK**, le nom est automatiquement reccourci aux 180 premiers caractères et la boîte de dialogue **Création de sous-programme** se ferme.

3. Si votre sous-programme utilise des arguments, (des repères pour les informations transmises au sous-programme) ajoutez-les un à un en cliquant sur le bouton **Ajouter argument...**. La boîte de dialogue **Modification de l'argument** apparaît. Pour plus

d'informations sur cette boîte de dialogue, voir la rubrique "Présentation de la boîte de dialogue Modification de l'argument".



Boîte de dialogue Édition de l'argument

4. Donnez un nom à votre argument en le tapant dans la case **Nom**.
5. Donnez une valeur par défaut à votre argument, en le tapant dans la zone **Valeur**. Le sous-programme utilise la valeur par défaut si aucune valeur n'est transmise dans le sous-programme à partir de l'instruction `CALLSUB`. Les valeurs valides d'arguments sont des valeurs numériques, des variables, des chaînes de texte et des noms d'éléments.
6. Si vous voulez donner une description à l'argument, entrez-la dans la case **Description**.
7. Cliquez sur **OK** dans la boîte de dialogue **Modification de l'argument** afin de le créer.
8. Répétez les étapes 3 à 7 pour chaque argument que vous voulez entrer dans votre sous-programme.
9. Cliquez sur le bouton **OK** dans la boîte de dialogue **Création sous-programme** pour terminer la création de votre sous-programme. Ce sous-programme apparaît dans la fenêtre de modification avec tous les arguments déterminés.
10. Terminez votre sous-programme en sélectionnant l'option de menu **Insérer | Commande contrôle de flux | Fin SP**. Ceci a pour effet de placer une commande `ENDSUB/` dans la fenêtre de modification, complétant ainsi le bloc de commande du sous-programme. Si vous voulez ajouter d'autres commandes de routine de mesure dans votre sous-programme vous devez le faire dans le bloc de commande du sous-programme, avant la commande `ENDSUB`.

Syntaxe d'un bloc de commande de sous-programme

Exemple de syntaxe de ligne de commande de la fenêtre Édition pour un bloc de commande de sous-programme :

```
<Nom>/SOUS-PROGRAMME
<A1> = <Arg1> : <Description>,
<A2> = <Arg2> : <Description>,
=
<Commandes>
FINSP/
```

SOUS-PROGRAMME/ est la commande utilisée pour démarrer le bloc de commande du sous-programme.

<Nom> = nom du sous-programme. Le nom est limité à 256 caractères ou moins. Si vous entrez 257 caractères ou plus, le nom est automatiquement raccourci à 256 caractères.

<A1> = le premier argument (ou variable locale) utilisé dans le sous-programme. Cette variable n'est généralement pas accessible en dehors du sous-programme.

<A2> = deuxième argument utilisé dans le sous-programme. Cette variable n'est généralement pas accessible en dehors du sous-programme. D'autres arguments peuvent être ajoutés, si nécessaire.

<Arg1> = La valeur par défaut du premier argument.

<Arg2> = La valeur par défaut du deuxième argument.

<Description> = La description de l'argument.

<Commandes> = D'autres commandes de la fenêtre de modification peuvent être insérées, selon les besoins, après les arguments et avant la commande FINSP.

FINSP/ est la commande utilisée pour terminer le bloc de commande du sous-programme.

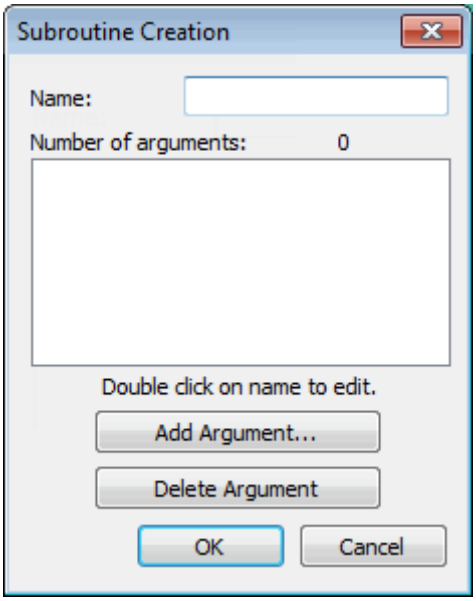
Exemple de bloc de commande de sous-programme

Par exemple, un sous-programme terminé, prenant les données de l'opérateur et les affichant dans le rapport, peut ressembler à ceci :

```
SOUS-PROGRAMME/PRENDRE_INFOS_OPÉRATEUR,
NOMOPÉ = <Opérateur> : NOM OPÉRATEUR,
DÉPLACEMENT = <Déplacement> : TEMPS DÉPLACEMENT,
=
COMMENTAIRE/RAPPORT,NOMOPÉ
COMMENTAIRE/RAPPORT,DÉPLACEMENT
FINSP/
```

Présentation de la boîte de dialogue Création de sous-programme

Pour ouvrir la boîte de dialogue **Création de sous-programme**, sélectionnez **Insérer | Commande de contrôle de flux | Sous-programme**.



Boîte de dialogue Création de sous-programme

Les options suivantes sont disponibles dans la boîte de dialogue **Création de sous-programme** :

Nom - Cette zone définit votre sous-programme. Il s'agit du nom que vous utiliserez quand vous appellerez ensuite le sous-programme ; aussi, si vous avez plusieurs sous-programmes dans une seule routine de mesure, chaque nom doit être unique.

Le nom est limité à 180 caractères ou moins. Si vous entrez 181 caractères ou plus dans la zone **Nom** le message d'erreur « Ligne trop longue » apparaît.

Quand vous cliquez sur le bouton **OK**, le nom est automatiquement raccourci aux 180 premiers caractères et la boîte de dialogue **Création de sous-programme** se ferme.

Nombre d'arguments - Cette liste montre les arguments pour le sous-programme que vous créez. Les arguments apparaissent dans cette zone sous cette forme :

```
<NOM> = <VALEUR> : <DESCRIPTION>
```

Par exemple, si l'un de ces arguments était nommé « Diamètre » avec une valeur par défaut de 3, votre argument dans cette liste peut être :

```
DIAMÈTRE = 3 : Le diamètre de l'alésage
```

PC-DMIS utilise la valeur par défaut quand une autre valeur n'est pas transmise depuis la commande `CALLSUB`.

Pour modifier un argument, double-cliquez simplement sur celui que vous voulez changer. La boîte de dialogue **Modification de l'argument** s'ouvre, vous permettant de faire les changements nécessaires. Pour plus d'informations sur cette boîte de dialogue, voir la rubrique "Présentation de la boîte de dialogue Modification de l'argument".

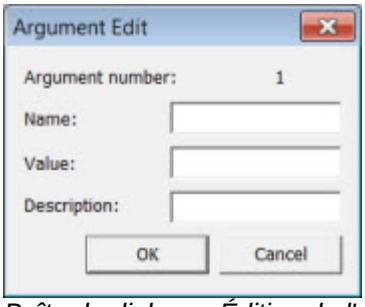
Ajouter argument - Ce bouton ajoute de nouveaux arguments à votre sous-programme. Cliquez sur ce bouton pour ouvrir la boîte de dialogue **Modification de l'argument**. Pour plus d'informations sur cette boîte de dialogue, voir la rubrique "Présentation de la boîte de dialogue Modification de l'argument".

Supprimer argument- Ce bouton vous permet de supprimer des arguments de votre sous-programme. Sélectionnez l'argument dans la liste, puis cliquez sur le bouton **Supprimer argument**.

Édition d'un sous-programme existant

Pour modifier un sous-programme existant, vous pouvez toujours utiliser le mode commande de la fenêtre de modification et modifier directement le sous-programme. Vous pouvez aussi accéder à la boîte de dialogue **Création de sous-programme** en plaçant votre curseur sur le sous-programme dans la fenêtre de modification et en appuyant sur F9. La boîte de dialogue **Création de sous-programme** s'ouvre. Pour des informations sur cette boîte de dialogue, voir la rubrique « Présentation de la boîte de dialogue Création de sous-programme ».

Présentation de la boîte de dialogue Édition de l'argument



Boîte de dialogue Édition de l'argument

La boîte de dialogue **Modification de l'argument** s'ouvre quand vous désirez créer ou modifier un argument dans les boîtes de dialogue **Création sous-programme** (**Insérer | Commande de contrôle de flux | Sous-programme**) ou **Appeler sous-programme** (**Insérer | Commande de contrôle de flux | Appeler sous-programme**).

La boîte de dialogue **Modification de l'argument** peut être utilisée dans les deux contextes suivants :

- Pour définir les arguments d'un sous-programme et leurs valeurs par défaut dans le bloc de commande `SUBROUTINE`.
- Pour définir les valeurs qui seront transmises dans le sous-programme à partir d'une commande `CALLSUB`.

Les options suivantes sont disponibles dans la boîte de dialogue **Modification de l'argument** :

Nom - Cette zone définit le nom de l'argument créé ou modifié.

Valeur - Cette zone définit la valeur de l'argument.

> Si vous créez ou modifiez la commande `SUBROUTINE`, il s'agit de la valeur par défaut utilisée si aucune valeur n'est transmise dans le sous-programme par l'instruction `CALLSUB`.
>
> Si vous créez ou modifiez la commande `CALLSUB`, il s'agit de la valeur par défaut transmise dans le sous-programme.
>
> Les valeurs valides peuvent être :
>
> - Numérique
> - Variable
> - Chaîne de texte - Les chaînes de texte doivent être incluses entre des guillemets double.
> - Nom d'élément - Le nom d'élément doit se trouver entre des accolades comme celles-ci : {F1}.

Description - Définit la description de l'argument dans le sous-programme. Cette description s'affiche en regard de l'argument dans le bloc de commande `SUBROUTINE` de la fenêtre de modification.

Appel d'un sous-programme

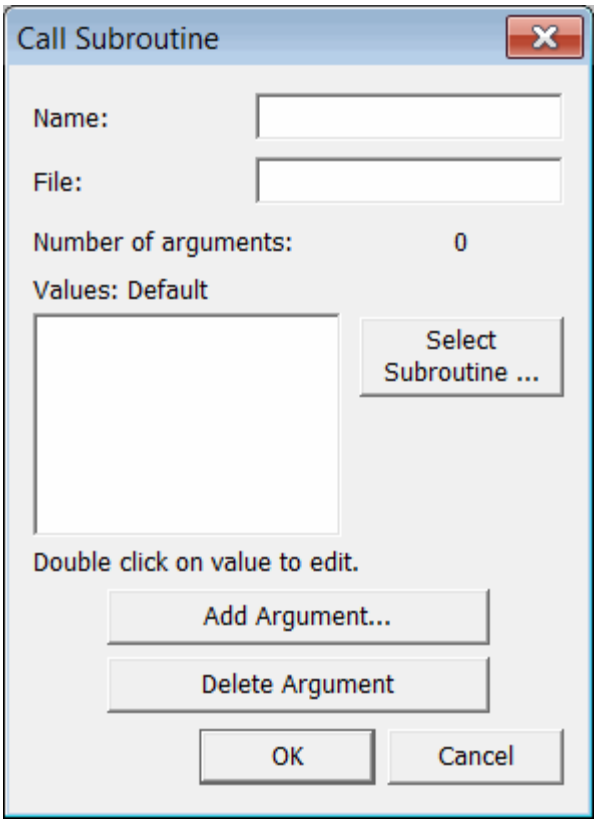
Pour appeler un sous-programme, vous devez insérer une commande APPELSP dans votre routine de mesure pour appeler une sous-routine existante dans la routine de mesure courante ou une sous-routine d'une routine de mesure externe.

Appel d'un sous-programme en tapant APPELSP

Vous pouvez insérer cette commande en entrant simplement CALLSUB dans la fenêtre de modification et en appuyant ensuite sur la touche TAB de votre clavier, là où vous voulez que la commande apparaisse dans la fenêtre de modification. Une fois la commande insérée, vous devez spécifier le nom du sous-programme, son emplacement s'il est dans une routine de mesure externe, de même que toute valeur à transmettre aux arguments disponibles. Voir "Transmission d'arguments dans un sous-programme" pour consulter des exemples de transmission d'arguments.

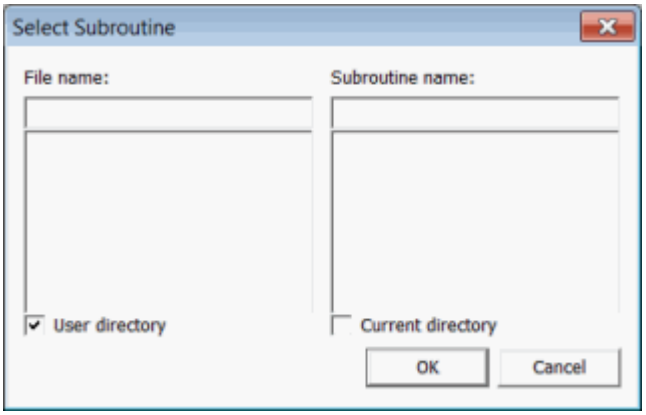
Appel d'un sous-programme en utilisant l'élément Appel sous-menu

1. Sélectionnez l'option **Insérer | Commande de contrôle de flux | Appeler sous-programme** dans le sous-menu. La ***boîte de dialogue*** *Appeler sous-programme* s'ouvre. Voir « Présentation de la boîte de dialogue Appeler sous-programme », pour plus d'informations la concernant.



Boîte de dialogue Appeler sous-programme

2. Cliquez sur le bouton **Sélectionner le sous-programme**. La ***boîte de dialogue*** *Sélectionner le sous-programme* s'ouvre.



Boîte de dialogue Sélectionner le sous-programme

3. Cochez la case **Répertoire utilisateur** et/ou la case **Répertoire courant**. Si la routine de mesure dont provient le sous-programme réside dans le répertoire indiqué pour la

recherche de sous-programmes, cochez la case **Répertoire utilisateur**. S'il provient du répertoire courant, cochez la case **Répertoire courant**. PC-DMIS répertorie toutes les routines de mesure disponibles pour la sélection.

4. Sélectionnez la routine de mesure qui contient le sous-programme souhaité. Vous voyez alors apparaître tous les sous-programmes associés à la routine de mesure sélectionnée dans la zone **Nom de sous-programme**.
5. Sélectionnez le sous-programme à appeler.
6. Cliquez sur le bouton **OK**. Les informations du sous-programme à appeler s'affichent dans les cases **Nom** et **Fichier** de la boîte de dialogue **Appeler sous-programme**.
7. Si vous voulez transmettre des informations dans le sous-programme, cliquez sur le bouton **Ajouter argument...** et utilisez la boîte de dialogue **Modifier argument** pour définir des arguments et des valeurs à transmettre. Voir "Présentation de la boîte de dialogue Modification de l'argument" pour plus d'informations la concernant. Voir "Transmission d'arguments dans un sous-programme" pour consulter des exemples de transmission d'arguments.
8. Cliquez à nouveau sur le bouton **OK**. La commande APPELSP s'affiche à l'emplacement sélectionné de la fenêtre de modification.

Syntaxe de la commande APPELSP

La syntaxe de la ligne de commande de la fenêtre Édition pour appeler un sous-programme est la suivante :

```
AS1 = APPELSP/<Nom>, <Fichier>:<Arg1>,<Arg2>,
```

AS1 = L'ID d'étiquette donné à la commande APPELSP.

<Nom> = Le nom du sous-programme à appeler.

<Fichier> = le parcours complet allant à la routine de mesure qui contient le sous-programme à appeler. Si ce champ est vide, PC-DMIS cherche le sous-programme dans la routine de mesure en cours.

<Arg1> = La valeur à transmettre au premier argument dans le sous-programme. Si ce champ est vide, la valeur par défaut définie pour le premier argument dans le sous-programme sera utilisée à la place.

<Arg2> = la valeur à transmettre au second argument dans le sous-programme. Si ce champ est vide, la valeur par défaut définie pour le second argument dans le sous-programme sera utilisée à la place. Cet exemple de syntaxe expose seulement deux arguments. D'autres arguments peuvent être transmis si votre sous-programme le nécessite.

Remarque : Il est conseillé de conserver un ensemble de pointeurs vers tous les objets faits pour le sous-programme dans votre commande APPELSP, de façon à pouvoir facilement vous y référer ultérieurement à l'aide de l'ID du sous programme. Pour en savoir plus sur les pointeurs, voir « Pointeurs », au chapitre « Utilisation d'expressions et de variables ».

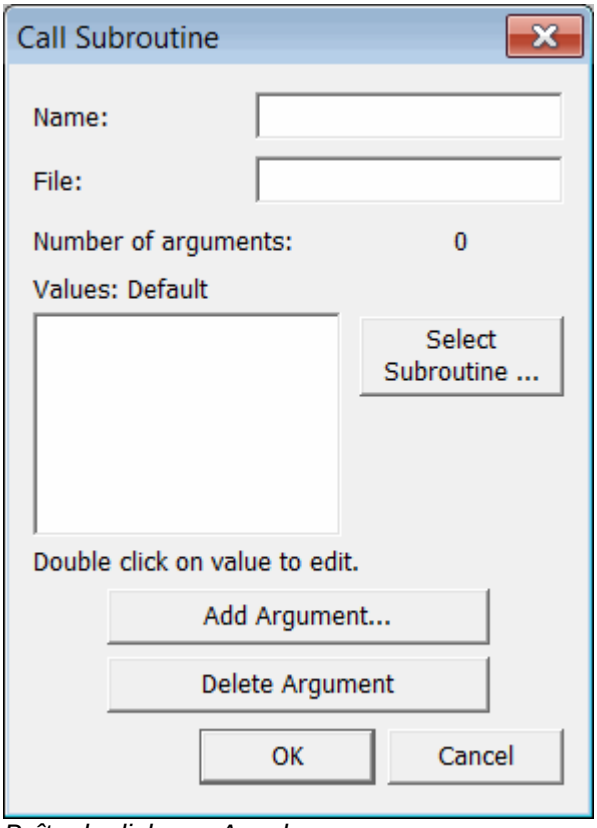
Exemple de commande APPELSP

```
CS1 =CALLSUB/GET_OPERATOR_INFO,D:\MEASROUTINES\V42SUBROUTINETEST.PRG:V1,V2,,
```

Cet exemple de commande CALLSUB CS1 appelle un sous-programme nommé GET_OPERATOR_INFO situé dans la routine de mesure V42SUBROUTINETEXT.PRG figurant dans le répertoire D:\MEASROUTINES\.

Elle transmet deux valeurs (dans ce cas, les variables V1 et V2) dans le sous-programme.

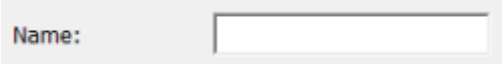
Présentation de la boîte de dialogue Appeler sous-programme



Boîte de dialogue Appeler sous-programme

Le tableau suivant décrit les options disponibles dans la boîte de dialogue **Appeler sous-programme**.

Zone **Nom**



La case **Nom** contient le nom du sous-programme sélectionné après avoir cliqué sur le bouton **Sélectionner le sous-programme...**

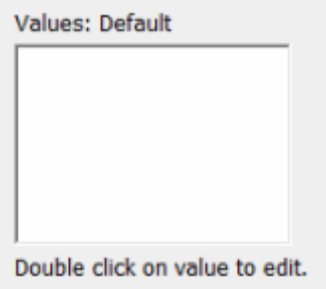
Le nom est limité à 180 caractères ou moins. Si vous entrez 181 caractères ou plus dans la zone **Nom** le message d'erreur « Ligne trop longue » apparaît.

Quand vous cliquez sur le bouton **OK**, le nom est automatiquement raccourci aux 180 premiers caractères et la boîte de dialogue **Appeler sous-programme** se ferme.

Zone **Fichier**

La case **Fichier** contient le chemin d'accès du répertoire au fichier du sous-programme appelé.

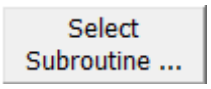
Zone **Valeurs**



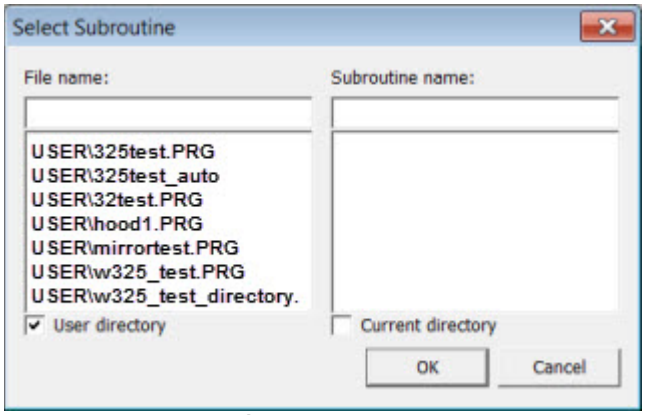
La zone **Valeurs** contient une liste des valeurs de chaque argument associé au sous-programme. Ces valeurs sont transmises au sous-programme quand celui-ci est exécuté.

Pour modifier ces valeurs, double-cliquez sur la valeur que vous voulez changer. La boîte de dialogue **Modification de l'argument** apparaît. Pour plus d'informations sur cette boîte de dialogue, voir la rubrique "Présentation de la boîte de dialogue Modification de l'argument".

Bouton **Sélectionner le sous-programme**



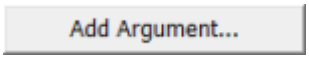
Le bouton **Sélectionner le sous-programme** affiche la boîte de dialogue **Sélectionner le sous-programme**.



Boîte de dialogue Sélectionner le sous-programme

Cette boîte de dialogue permet d'appeler des sous-programmes créés précédemment en effectuant une recherche dans le répertoire utilisateur ou le répertoire courant. Voir « Appel d'un sous-programme », pour obtenir des informations sur la boîte de dialogue **Sélectionner le sous-programme**.

Bouton **Ajouter argument**



Ce bouton définit une valeur à transmettre aux arguments pour le sous-programme.

Bouton **Supprimer argument**

Ce bouton vous permet de supprimer les arguments de la zone **Valeurs**. Sélectionnez la valeur affichée et cliquez sur le bouton **Supprimer argument**. L'argument associé à cette valeur est alors supprimé.

Ajout d'un nouvel argument à l'aide de la boîte de dialogue Appeler sous-programme :

Ajout d'un nouvel argument dans votre commande `CALLSUB` pour passer dans le sous-programme :

1. Dans la boîte de dialogue **Appeler sous-programme**, cliquez sur le bouton **Ajouter argument** pour ouvrir la boîte de dialogue **Modification de l'argument**.
2. Cliquez dans la case **Valeur**.
3. Dans la zone **Valeur**, entrez la valeur de l'argument.
4. Cliquez sur **OK**.

Modification d'arguments existants à l'aide de la boîte de dialogue Appeler sous-programme :

Édition d'un argument existant dans votre commande CALLSUB :

1. Dans la boîte de dialogue **Appeler sous-programme**, double-cliquez sur la valeur de l'argument que vous voulez changer. Une zone de valeur s'affiche et présente la valeur par défaut du sous-programme appelé.
2. Tapez une nouvelle valeur.
3. Cliquez sur le bouton **OK**.

Voir "Création d'un sous-programme" et "Modification d'un sous-programme existant" pour plus d'informations sur l'édition ou la création d'arguments de sous-programme.

Pour supprimer des arguments d'une commande d'appel de sous-programme :

1. Placez votre curseur sur la commande APPELSP.
2. Appuyez sur F9 pour accéder à la boîte de dialogue **Appeler sous-programme**.
3. Dans la liste d'arguments, sélectionnez l'un des arguments.
4. Cliquez sur le bouton **Supprimer argument**.
5. Cliquez sur **OK**.

Vous pouvez également supprimer un argument directement dans le texte de la fenêtre Édition. Pour ce faire :

1. Passez PC-DMIS en mode commande.
2. Placez le curseur sur la commande APPELSP, puis appuyez sur la touche TAB jusqu'à la mise en surbrillance de l'argument désiré.
3. Entrez "del" pour supprimer l'argument. Le fait d'appuyer sur la touche Suppr ou Retour ne supprime pas vraiment l'argument : i lest simplement remplacé par un argument vide.

Utilisation d’instructions APPEL SP en mode maître/esclave

Si vous affectez une instruction CALLSUB *au bras 1*, toutes les commandes dans le sous-programme seront également affectées au bras1 à l'appel du sous-programme.

Si vous affectez une instruction CALLSUB *au bras 2*, toutes les commandes dans le sous-programme seront également affectées au bras2 à l'appel du sous-programme.

Si vous marquez une instruction APPELSP *pour une exécution sur les deux bras*, PC-DMIS laisse les marquages du sous-programme tels que configurés à l'origine.

Si un sous-programme contient une commande *DÉPLACER/SYNC* et que vous affectez l'instruction APPELSP au bras1 ou au bras2, au moment de l'exécution PC-DMIS affiche un message d'erreur signalant une commande invalide et l'échec de l'appel du sous-programme.

Pour plus d'informations sur l'affectation d'une commande en vue de son exécution sur un bras spécifique, voir la rubrique « Affectation d'une commande à un bras », au chapitre « Utilisation du mode bras multiple »

Exemples de sous-programmes

Considérez les informations des rubriques suivantes concernant des exemples de transmissions d'arguments et de sous-programmes, en général.

- Transmission d'arguments dans un sous-programme
- Exemple de code d'un sous-programme

Transmission d'arguments dans un sous-programme

Les types d'arguments qui peuvent être transmis à un sous-programme sont des valeurs numériques, des variables, des chaînes de texte et des noms d'éléments. Pour transmettre des valeurs à des arguments, tapez la valeur dans la case **Valeur** de la boîte de dialogue **Modification de l'argument** ou directement dans une commande CALLSUB insérée en mode commande.

Pour des détails sur la façon de procéder, voir "Création d'un sous-programme" et "Présentation de la boîte de dialogue Modification de l'argument".

Transfert de variables dans un sous-programme

Les arguments pouvant renvoyer des données sont des variables. Lorsqu'une variable est utilisée en tant qu'argument d'un sous-programme, toutes les modifications affectant la variable correspondante dans le sous-programme sont retournées et deviennent la valeur de la variable transmise.

Exemple de transmission de variables :

Cet exemple illustre comment la valeur d'une variable, utilisée en tant qu'argument d'un sous-programme, est modifiée et retournée à partir d'un sous-programme :

La variable V1 reçoit la valeur 6 :

```
ASSIGN/ V1 = 6.
```

Un appel de sous-programme passe V1 en tant que premier argument :

```
CS1 =APPELSP/MONSP,:V1,,,
```

Le sous-programme est défini comme suit :

```
SOUS-PROGRAMME/MONSP,
A1 = 0 : PREMIER ARGUMENT,
=
ASSIGN/ A1 = A1 + 1
FINSP/
```

A1 est le nom du premier argument, ainsi lorsque l'appel est effectué, A1 a la même valeur que V1 au moment de l'appel, c'est-à-dire 6.

Le sous-programme contient cette instruction :

```
ASSIGN/ A1 = A1 + 1.
```

Ceci incrémente la valeur de A1 à 7.

Puis le sous-programme finit avec la commande `FINSP/`.

Le flux de l'exécution retourne à l'instruction directement après la commande `CALLSUB`. Lors du retour de l'exécution, toutes les variables qui étaient utilisées comme arguments (V1 dans le cas présent) sont mises à jour avec la valeur des variables correspondantes dans le sous-programme (A1 dans l'exemple). V1 a désormais la valeur 7. La valeur a été transmise en retour du sous-programme.

Transmission de valeurs de nombres dans un sous-programme

Des arguments peuvent aussi prendre des caractères numériques.

Exemple de transmission de valeurs de nombres :

Cet exemple montre comment transmettre des valeurs de nombres dans un sous-programme. Il transfère deux nombres, puis les ajoute l'un à l'autre.

```
CS1 = APPELSP/SOM_NOMBRES,:,,,
CS2 = APPELSP/SOM_NOMBRES, : 5,10,,

SOUS-PROGRAMME/SOM_NOMBRES,
NOM1 = 1 : PREMIER NOMBRE,
NOM2 = 1 : DEUXIÈME NOMBRE,
=
COMMENT/OPER,NO,« La somme du premier nombre " + NOM1 + ", plus le deuxième nombre,
" + NOM2 + ", est : »
,NOM1 + NOM2
FINSP/
```

Dans la première commande CALLSUB (CS1) aucune valeur de nombre n'est transmise au sous-programme. Les nombres par défaut, 1 pour NUM1 et 1 pour NUM2 sont utilisés à la place et la somme ainsi générée est 2.

Dans la deuxième commande CALLSUB (CS2), deux valeurs de nombre sont transmises (5 et 10). NUM1 équivaut à 5 et NUM2 à 10, avec une somme générée de 15.

Transfert de chaînes de texte dans un sous-programme

Des arguments peuvent aussi prendre une chaîne de texte. Pour transmettre une chaîne de texte, assurez-vous que les caractères alphanumériques sont placés entre des guillemets.

Exemple de transfert de chaînes de texte :

Cet exemple montre comment transmettre des valeurs de chaînes dans un sous-programme. Il transmet deux valeurs de chaînes aux deux paramètres et les affiche ensuite dans le rapport

```
CS1 = APPELSP/PRENDRE_INFOS_OPÉRATEUR,:« BOB JONES »,« BONJOUR »,,

            ...

            SOUS-PROGRAMME/PRENDRE_INFOS_OPÉRATEUR,
            NOMOPÉ = <Opérateur> : NOM OPÉRATEUR,
            DÉPLACEMENT = <Déplacement> : TEMPS DÉPLACEMENT,
            COMMENTAIRE/RAPPORT,NOMOPÉ
            COMMENTAIRE/RAPPORT,DÉPLACEMENT
            FINSP/
```

Le premier argument `OPNAME` reçoit la valeur "BOB JONES", le second argument `SHIFT` reçoit la valeur "BONJOUR". Les commandes COMMENT envoient ensuite les chaînes de transmission au rapport d'inspection.

Transfert de noms d'éléments dans un sous-programme

Les noms d'éléments sont transmis entre accolades, par exemple : {F1} est utilisé pour appeler l'élément désigné par F1 dans la fenêtre de modification. De plus, lors du transfert du nom d'élément, le sous-programme a totalement accès à cet élément.

Exemple de transmission de noms d'éléments :

Cet exemple transmet le nom d'élément PNT1 au sous-programme, permettant au sous-programme un accès complet à l'élément. Si aucune valeur n'est transmise ensuite, le nom d'élément par défaut, qui est F1, est utilisé. Ensuite, ce sous-programme demande une nouvelle valeur et change la valeur théorique X pour l'élément.

```
CS1 = APPELSP/CHANGER_THÉO_X, : {PNT1},,

SOUS-PROGRAMME/CHANGER_THÉO_X,
ÉLÉMENT1 = {F1} : NOM D'ÉLÉMENT TRANSMIS,
=
C1 =COMMENTAIRE/ENTRÉE,NO,« ÉLÉMENT TRANSMIS : »
,ÉLÉMENT1
,« La valeur théorique X est : " + ÉLÉMT1.TX
,« Entrez une nouvelle valeur théorique X : »
ASSIGN/ÉLÉMT1.TX = INPUTC1.
COMMENTAIRE/OPER,NO,« Élément mis à jour à » + ÉLÉMT1.TX
FINSP/
```

Comme le sous-programme a accès à l'élément transmis, l'instruction `ASSIGN/ÉLÉMT1.TX = C1.INPUT` modifie la valeur X théorique de l'élément PNT1. La valeur X théorique de PNT1 est définitivement modifiée.

Exemple de code d'un sous-programme

L'exemple de code suivant offre à l'opérateur la possibilité de modifier les valeurs théoriques X, Y, et Z d'un élément après la mesure de celui-ci. Les exécutions suivantes utiliseront alors les valeurs théoriques mises à jour.

```
PNT1 =GENERIC/POINT,DEPENDENT,CARTESIAN,$
NOM/XYZ,<5,10,15>,$
MEAS/XYZ,<7,12,17>,$
NOM/IJK,<0,0,1>,$
MEAS/IJK,<0,0,1>
```

```
C1 =COMMENT/YESNO,NO,FULL SCREEN=NO,AUTO-CONTINUE=NO,
Souhaitez-vous modifier les valeurs théoriques de PNT1 ?
IF/INPUTC1.==« OUI »
CS1=CALLSUB/CHANGETHEO,:,
END_IF/
COMMENT/OPER,NO,FULL SCREEN=NO,AUTO-CONTINUE=NO,
Les valeurs XYZ théoriques et réelles pour PNT1 sont les suivantes :
"Theo X= " + PNT1.TX
"Theo Y= " + PNT1.TY
"Theo Z= " + PNT1.TZ
--------------------
"Actl X= " + PNT1.X
"Actl Y= " + PNT1.Y
"Actl Z= " + PNT1.Z
ROUTINE/END
SOUSPPROGRAMME/CHANGERTHÉO,
POINT1 = {PNT1} : ,
=
INFODIM/PNT1;ICÔNE,IDDIM,IDÉLÉM,VERT,HORIZ, ,$
EN-TÊTES, ;MESU, , , , , , , ,
C2 =COMMENT/INPUT,NO,FULL SCREEN=NO,
Tapez la nouvelle valeur X théorique pour PNT 1.
« Sa valeur réelle est : » + PNT1.TX
ASSIGN/PNT1.TX=INPUTC2.
C3 =COMMENT/INPUT,NO,FULL SCREEN=NO,
Tapez la nouvelle valeur Y théorique pour PNT 1.
« Sa valeur réelle est » + PNT1.TY
ASSIGN/PNT1.TY=INPUTC3.
C4 =COMMENT/INPUT,NO,FULL SCREEN=NO,
Tapez la nouvelle valeur Z théorique pour PNT 1.
« Sa valeur réelle est » + PNT1.TZ
ASSIGN/PNT1.TZ=INPUTC4.
FINSP/
```

Explication du code exemple

C1=COMMENTAIRE/OUINON
Cette ligne enregistre et stocke la réponse affirmative ou négative de l'utilisateur.

SI/INPUTC1.**==« OUI »**
Cette ligne constitue l'expression. Elle permet de vérifier que l'entrée de commentaire 1 est OUI. S'il s'agit bien de OUI, l'instruction SI est VRAIE, les instructions après SI sont exécutées et, selon cet exemple, l'élément `PNT1` est mesuré. Si la réponse est NON, le programme passe directement à l'instruction FIN SI.

`CS1 = CALLSUB/CHANGETHEO,:,`
Cette ligne appelle le sous-programme nommé CHANGETHEO. Le flux de la routine de mesure saute alors à la ligne `SUBROUTINE/CHANGETHEO`.

`SUBROUTINE/CHANGETHEO`
Cette ligne initialise le sous-programme CHANGETHEO. Le flux de la routine de mesure continue d'exécuter le code présent entre cette ligne et la ligne `ENDSUB/`.

`POINT1 = {PNT1} : ,`
Il s'agit du seul argument du sous-programme. Il permet au sous-programme d'accéder aux informations via l'élément PNT1.

`C2=COMMENTAIRE/ENTRÉE, C3=COMMENTAIRE/ENTRÉE, C4=COMMENTAIRE/ENTRÉE`
Ces commentaires d'entrée permettent tous d'enregistrer les nouvelles valeurs théoriques X, Y et Z entrées par l'utilisateur et de les stocker dans INPUTC2., INPUTC3. et INPUTC4. respectivement.

`ASSIGN/PNT1.TX = INPUT`
C2.Cette ligne enregistre la valeur théorique X à partir de ENTRÉEC2. et l'affecte à la variable `PNT1.TX`. `PNT1.TX` est une variable PC-DMIS qui contient la valeur théorique X (indiquée par TX) du point doté de l'étiquette ID de PNT1.

`ASSIGN/PNT1.TY = INPUT`
C3. Cette ligne enregistre la valeur théorique Y à partir de ENTRÉEC3. et l'affecte à la variable `PNT1.TY`. `PNT1.TY` est une variable PC-DMIS qui contient la valeur théorique Y (indiquée par TY) du point doté de l'étiquette ID de PNT1.

`ASSIGN/PNT1.TZ = C4.INPUT`
Cette ligne enregistre la valeur théorique Z à partir de ENTRÉEC4. et l'affecte à la variable `PNT1.TZ`. `PNT1.TZ` est une variable PC-DMIS qui contient la valeur théorique Z (indiquée par TZ) du point doté de l'étiquette ID de PNT1.

FINSP/
Cette ligne marque la fin du sous-programme et le retour du flux de la routine de mesure à la ligne immédiatement consécutive à celle de l'appel du sous-programme. Dans le cas présent, il s'agit de l'instruction `END_IF/`. Le flux de la routine de mesure passe alors au commentaire de l'opérateur suivant qui affiche les valeurs théoriques et réelles X, Y et Z, puis la routine de mesure se termine par la commande `ROUTINE/END`.

Fin d'une routine de mesure

L'option de menu **Insérer | Commande de contrôle de flux | Fin de routine** insère une commande `ROUTINE/END` dans la fenêtre de modification. Chaque fois que PC-DMIS rencontre cette commande lors de l'exécution d'une routine, il interrompt immédiatement cette dernière.

Cette commande s'avère utile pour mettre fin à une routine de mesure plus tôt qu'habituellement, en fonction de conditions définies.