Table des matières

Utilisation d'expressions et de variables 1
Utilisation d'expressions et de variables : introduction 1
Utilisation d'expressions dans une routine de mesure 1
Affichage de valeurs d'expressions 2
Conservation de valeurs d'expressions uniquement 2
Utilisation d'expressions avec un branchement 2
Utilisation d'expressions avec un fichier E/S 2
Création d'expressions via la construction d'expressions 3
Utilisation de variables dans des expressions 8
Compréhension des composantes d’expressions 9
Littéraux 9
Références 10
Variables 16
Structures 19
Pointeurs 20
Tableaux 22
Opérateurs pour les expressions 30
Priorité 31
Fonctions 31
Liste de fonctions 31
Exemple de droite créée à partir d'un segment de scan 63
Contrainte d'opérande 65
Expressions d'identification 67
Accès aux propriétés d'un objet de rapport 70
Accès aux informations à partir d'un cercle minimum de scan construit 72

Utilisation d'expressions et de variables

Utilisation d'expressions et de variables : introduction

Une expression est une condition définie par l'utilisateur et employée dans les commandes de contrôle de flux de PC-DMIS. Les instructions de contrôle de flux permettent de tester ces conditions dans votre routine de mesure, de vérifier si la condition est remplie ou non et de déterminer l'action à exécuter par PC-DMIS.

Les expressions jouent un rôle important dans l'accomplissement de vos tâches spécifiques par PC-DMIS. L'utilisation d'expressions conjointement avec les commandes de contrôle de flux vous permet de décupler encore plus la capacité des puissantes fonctionnalités de PC-DMIS.

Ce chapitre explique comment créer et utiliser des expressions dans la fenêtre de modification de PC-DMIS. Lorsque vous employez des expressions, vous devez passer la fenêtre de modification de PC-DMIS en mode commande. Vous pouvez ainsi afficher directement le code de la fenêtre de modification.

Le présent chapitre regroupe les rubriques suivantes :

- Utilisation d'expressions dans une routine de mesure
- Création d'expressions via le constructeur d'expressions
- Utilisation de variables dans des expressions
- Compréhension des composantes d'expressions
- Accès aux propriétés d'un objet de rapport
- Accès aux informations à partir d'un cercle minimum de scan construit

Pour plus d'informations sur les expressions de génération de rapports, voir "À propos des expressions de rapport" au chapitre "Rapport de résultats de mesure".

Utilisation d'expressions dans une routine de mesure

La fenêtre Édition de PC-DMIS autorise la saisie d'expressions dans la plupart de ses champs d'édition. En mode commande, les zones d'édition sont généralement celles en surbrillance jaune lorsque vous appuyez sur la touche TAB dans la fenêtre Édition. En revanche, les zones qui changent le type de l'élément n'acceptent pas d'expressions.

Exemple :la zone d'un élément automatique qui en spécifie le type (point de surface, cercle automatique, logement oblong automatique, etc.) n'accepte pas d'expressions.

Les sous-rubriques de la présente rubrique exposent en détail les expressions disponibles.

Affichage de valeurs d'expressions

Pour consulter la valeur d'une expression, maintenez le curseur de la souris sur l'expression pendant au moins une seconde. L'expression est évaluée et s'affiche dans une petite fenêtre contextuelle jaune, sa valeur actuelle apparaissant juste au-dessous du curseur.

Conservation de valeurs d'expressions uniquement

Pour résoudre immédiatement une expression dans la fenêtre Édition et n'en conserver ainsi que la valeur :

1. Sélectionnez le texte de l'expression dans la fenêtre Édition.
2. Faites précéder ce texte du caractère ` (accent grave).

Exemple : imaginez que vous entrez l'expression "`**1/7**" dans une zone numérique. L'expression est immédiatement résolue et seule la valeur (**0,143**) apparaît dans la zone.

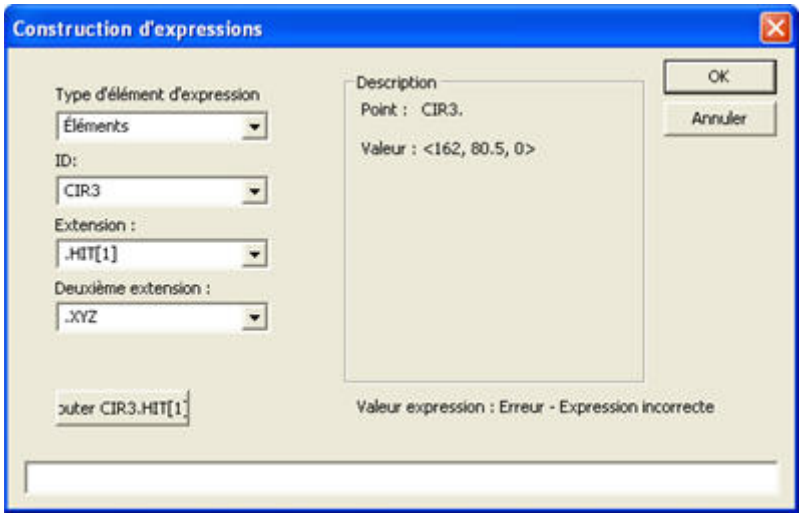
Utilisation d'expressions avec un branchement

Les commandes de contrôle de flux utilisent des expressions pour déterminer le déroulement de l'exécution de la routine. Voir le chapitre "Branchement à l'aide du contrôle de flux" pour plus d'informations sur les possibilités de branchement.

Utilisation d'expressions avec un fichier E/S

L'écriture ou la lecture de données dans un fichier de données externe requiert souvent l'utilisation de variables et autres expressions afin de gérer et de stocker ou d'afficher ces données de manière efficace. Voir le chapitre "Utilisation d'entrées/sorties de fichier" pour plus d'informations.

Création d'expressions via la construction d'expressions



Boîte de dialogue Construction d'expressions

PC-DMIS vous permet de créer et d'ajouter des expressions dans la fenêtre de modification à l'aide du clavier ou de l'interface plus conviviale de la boîte de dialogue **Construction d'expressions**. L'option de menu **Modifier | Expression** ouvre la boîte de dialogue **Construction d'expressions**.

Cette boîte de dialogue vous permet de créer une expression et de l'insérer dans un champ à éditer. Appuyer sur la touche F2 quand que le curseur est sur une zone acceptant des expressions permet également l'ouverture de la boîte de dialogue **Construction d'expressions**.

La boîte de dialogue **Construction d'expressions** répertorie tous les types d'opérateurs et de fonctions disponibles pour les expressions.

Création d'une expression à l'aide du clavier

Pour créer une expression en la saisissant directement dans la fenêtre Édition :

1. Ouvrez la fenêtre de modification (**Afficher | Fenêtre de modification**).
2. Passez-la en mode commande.
3. Appuyez sur la touche TAB pour placer le curseur dans la zone à éditer où vous voulez insérer l'expression. Les zones en surbrillance jaune sont « à éditer ».
4. Entrez l'expression.

Création d'une expression via la boîte de dialogue Construction d'expressions

Remarque : vous devez être en mode commande pour que l'option Expression soit activée.

Pour entrer une expression à l'aide de la boîte de dialogue Construction d'expressions (**Modifier | Expression**) :

1. Ouvrez la fenêtre de modification (**Afficher | Fenêtre de modification**).
2. Passez la fenêtre de modification en mode commande (**Afficher | Mode commande**).
3. Placez le curseur dans la zone à éditer où vous voulez insérer l'expression.
4. Appuyez sur la touche F2 quand le curseur est sur une zone acceptant les expressions. La boîte de dialogue **Construction d'expressions** s'ouvre. La boîte de dialogue **Construction d'expressions** répertorie tous les types d'opérateurs, d'opérandes et de fonctions. Les éléments suivants peuvent être référencés via cette boîte de dialogue :

    - Types disponibles d'expressions
    - Variables
    - Éléments
    - Dimensions
    - Alignements
    - Commentaires

5. Sélectionnez le type d'élément d'expression dans la première liste déroulante. Selon votre choix, d'autres zones de listes modifiables apparaissent.
6. Sélectionnez l'ID désiré dans la liste déroulante **ID**.
7. Sélectionnez une extension dans la liste déroulante **Extension**.
8. Sélectionnez une autre extension dans la liste déroulante **Deuxième extension**. Si l'expression est utilisable, le bouton **Ajouter** devient disponible.
9. Cliquez sur le bouton **Ajouter**. L'expression apparaît dans une zone d'édition.
10. Cliquez sur le bouton **OK**. L'expression apparaît maintenant dans la fenêtre Édition à l'emplacement du curseur.

Remarque : vous pouvez aussi ouvrir la boîte de dialogue **Construction d'expressions** à partir d'autres boîtes de dialogue :

- La boîte de dialogue **If Expression** - Sélectionnez **Insérer | Commande de contrôle de flux | If Goto**. Cliquez sur le bouton **Expression**.

- La boîte de dialogue **Affectation** - Sélectionnez **Insérer | Affectation**. Cliquez sur le bouton **Affecter à** ou **Affecter de**.

Une fois l'expression créée, PC-DMIS insère automatiquement l'expression à la position correcte suivante, dans la fenêtre de modification.

Vérification de l'exactitude d'une expression

Lorsque le curseur n'est plus dans la zone où vous avez ajouté l'expression, PC-DMIS tente de vérifier que l'expression est correcte. En cas de problème, un message d'erreur indique un chiffre

non valide ou le texte de l'expression s'affiche en rouge. Les expressions qui font référence à des objets n'existant pas s'affichent également en rouge.

La vérification de l'expression ayant lieu lorsque vous quittez la zone, celle qui s'affiche en rouge en raison d'une référence à un objet non existant (par exemple, CIRCLE1.X) reste rouge même si le nouvel objet (par exemple, CIRCLE1) est ajouté. La zone reste rouge tant que l'expression n'est pas testée à nouveau.

Pour vérifier à nouveau que l'expression est correcte :

1. Placez le curseur dans la zone de l'expression.
2. Appuyez sur la touche F2 . La boîte de dialogue **Construction d'expressions** apparaît de nouveau. Toute modification apportée à l'expression apparaît dans la zone d'édition.
3. Appuyez sur la touche ENTRÉE pour fermer la boîte de dialogue.

Type d'élément d'expression

La liste déroulante Type d'élément d'expression de la boîte de dialogue Construction d'expressions (**Modifier | Expression**) répertorie les différents types d'éléments d'expressions pouvant être utilisés dans des expressions. Elles incluent :

- Fonctions
- Opérateurs
- Alignements
- Commentaires
- Dimensions
- Éléments
- Variables

ID

La liste déroulante ID de la boîte de dialogue **Construction d'expressions** (**Modifier | Expression**) répertorie la série d'éléments disponibles en fonction du type d'élément d'expression sélectionné dans la liste déroulante Type d'élément d'expression.

Par exemple :

- Lorsque vous sélectionnez **Fonctions et opérateurs** dans la liste déroulante **Type d'élément d'expression**, la liste déroulante **ID** contient la liste des fonctions et des opérateurs disponibles.
- Lorsque vous sélectionnez des éléments dans la liste déroulante **Type d'éléments d'expression**, la liste déroulante **ID** affiche les ID des éléments dans la routine de mesure.

Extension

La liste déroulante **Extension** dans la boîte de dialogue Construction d'expression (**Modifier | Expression**) devient disponible lorsque l'élément sélectionné dans la liste déroulante **ID** requiert l'ajout d'une extension afin de formuler un élément d'expression utilisable. La liste déroulante **Extension** affiche les extensions disponibles selon l'élément sélectionné dans la liste déroulante ID.

Exemple : vous sélectionnez un élément dans la liste déroulante **ID**. La liste des extensions disponibles permettant de référencer les données de cet élément (telles que "X", "Y", "Z", "Diam", "Longueur", etc.) sont alors répertoriées dans la boîte de dialogue **Extension**.

Les extensions possibles incluent ces types de données mesurées ou théoriques :

- Tout – Toutes les valeurs de l'élément sont attribuées à la variable. Voir l'exemple ci-dessous.
- X – Valeurs mesurées X des palpages
- Y – Valeurs mesurées Y des palpages
- Z – Valeurs mesurées Z des palpages
- XYZ – Valeurs mesurées XYZ des palpages
- I – Valeurs mesurées I des palpages
- J – Valeurs mesurées J des palpages
- K – Valeurs mesurées K des palpages
- IJK – Valeurs mesurées IJK des palpages

- TX – Valeurs théoriques X des palpages
- TY – Valeurs théoriques Y des palpages
- TZ – Valeurs théoriques Z des palpages
- TXYZ – Valeurs théoriques XYZ des palpages
- TI – Valeurs théoriques I des palpages
- TJ – Valeurs théoriques J des palpages
- TK – Valeurs théoriques K des palpages
- TIJK – Valeurs théoriques IJK des palpages

Exemple :

```
F1=GENERIC/POINT,DEPENDENT,CARTESIAN,$
      NOM/XYZ,<8,9,10>,$
      MEAS/XYZ,<7.98,8.98,9.98>,$
      NOM/IJK,<1,0,0>,$
      MEAS/IJK,<1,0,0>
```

```
ASSIGN/MYFEATURE = F1.ALL
ASSIGN/V1 = MYFEATURE.X
ASSIGN/V2 = MYFEATURE.TX
```

V1 contient la valeur 7,98

V2 contient la valeur 8

Deuxième extension

La liste déroulante **Deuxième extension** ne devient disponible que si l'élément choisi dans la liste déroulante **Extension** impose l'ajout d'une seconde extension afin de formuler un élément d'expression utilisable.

Exemple : imaginez que vous voulez faire référence à la valeur nominale de l'axe d'emplacement X d'une dimension nommée "D1".

Pour ce faire :
sélectionnez **D1** dans la liste déroulante **ID**.
Sélectionnez **X** dans la liste déroulante **Extension**
Sélectionnez **Nom** dans la liste déroulante **Deuxième extension**.

Bouton Ajouter

Lorsque vous sélectionnez un élément d'expression utilisable ou complet à l'aide des listes déroulantes, le bouton **Ajouter** devient disponible. Ce bouton affiche le texte à ajouter à l'expression.

Si vous choisissez par exemple les éléments suivants :

- Dimensions dans la liste **Type d'élément d'expression**
- D1 dans la liste **ID**
- X dans la liste **Extension**
- Nom dans la liste **Deuxième extension**

Le bouton **Ajouter** debient actif et affiche le texte suivant : **Add D1.X.NOM**.

Lorsque vous cliquez sur le bouton **Ajouter**, le texte apparaît dans la zone d'édition, dans la partie inférieure de la boîte de dialogue.

Remarque : Lorsque vous cliquez sur le bouton **OK**, le texte de la zone d'édition est ajouté dans la fenêtre de modification, dans la zone d'expression où se trouve le curseur. Si vous sélectionnez un élément à partir de la zone d'expression de la fenêtre de modification et que le texte à ajouter comporte des parenthèses, alors l'élément sélectionné est placé entre les parenthèses du texte ajouté.

Zone d'édition

Au bas de la boîte de dialogue **Construction d'expressions** (**Modifier | Expression**) se trouve une zone d'édition affichant l'expression en cours. Vous pouvez entrer directement l'expression dans la zone ou utiliser le bouton **Ajouter**.

Zone Description

La boîte de dialogue **Construction d'expressions** (**Modifier | Expression**) comporte également une zone **Description** fournissant des informations sur les éléments sélectionnés dans les listes déroulantes. Un zone en regard du bouton **Ajouter** affiche également la valeur actuelle de l'expression.

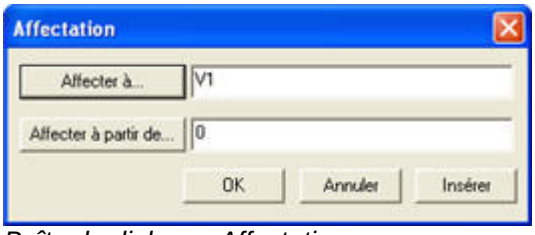
Remarque : les expressions non valides ont la valeur 0.

Utilisation de variables dans des expressions

Les variables sont des objets qui conservent des valeurs. Elles font référence aux opérandes entier, réel, chaîne ou point. Elles sont indispensables à l'utilisation d'expressions. Une variable possède un nom et une valeur. Le nom permet d'accéder à la valeur de la variable. Le nom ne change pas, seule la valeur est modifiable. Vous affectez une valeur à une variable à l'aide de la commande `ASSIGN/`.

Par exemple, l'instruction `ASSIGN/V1 = 2` crée une variable nommée V1 avec la valeur 2. `ASSIGN/V2 = V1 + 2` accepte la valeur V1. Si la valeur de V1 était encore 2 au moment de l'exécution de cette instruction d'affectation, V2 aurait une valeur de 4.

Pour plus d'informations sur les variables, voir « Variables ».



Boîte de dialogue Affectation

L'option de menu **Insérer | Affectation** permet d'afficher la boîte de dialogue **Affectation**. Cette boîte de dialogue permet d'affecter une valeur à une variable ou à un élément de données d'un élément, d'une dimension ou d'un alignement de routine de mesure. L'utilisation de la commande d'affectation requiert une connaissance de base des expressions de PC-DMIS.

Bouton **Affecter à** - Ce bouton permet de spécifier la variable qui reçoit la valeur calculée dans la zone **Affecter à partir de**. Les informations choisies à l'aide du bouton **Affecter à** sont placées dans la zone **Affecter à**. Cette valeur peut correspondre au nom d'une variable ou une référence à un élément de données d'un élément, d'une dimension ou d'un alignement.

Le terme « évalué » renvoie au résultat obtenu après la résolution d'une expression mathématique pour une valeur donnée.

Bouton **Affecter à partir de** - Ce bouton permet de placer la valeur affectée dans la case **Affecter à partir de**. Si cette case contient une expression, celle-ci est évaluée au moment de l'exécution et le résultat ou la valeur du calcul est affecté à l'objet spécifié dans la case **Affecter à**.

Bouton **Insérer** - Ce bouton permet d'insérer une commande d'affectation dans la routine de mesure, tout en conservant la boîte de dialogue **Affectation** ouverte. Vous pouvez insérer une série de commandes d'affectation sans fermer la boîte de dialogue.

Compréhension des composantes d'expressions

Les expressions ont ces types d'opérandes :

- Entiers
- Nombres réels
- Chaînes
- Points
- Pointeurs d'élément
- Tableaux
- Fonctions

Ces onglets sont décrits en détail ci-après :

Types d'opérandes

Les opérandes peuvent prendre plusieurs formes :

- Littéraux
- Références
- Variables
- Structures
- Pointeurs
- Tableaux

Littéraux

***Entiers :** 1, -6, 209

Valeurs réelles : 1, -6, 2.4, -0.1, 345,6789

Chaînes : 3Hello World", "47", "CIRCLE 1"

Points : vous ne pouvez pas utiliser de représentation littérale pour les points. Ces derniers peuvent toutefois s'obtenir à partir d'autres littéraux avec la fonction MPOINT : par exemple, `MPOINT(0,0,1)`, `MPOINT(2.2, 3.1, 4.0)`.

Pointeur :nom d'un élément placé entre crochets, comme : `{CIR1}, {LIN2}, {F3}`

Tableaux : aucune représentation de littéral n'est disponible pour les tableaux. Il est toutefois possible de créer des tableaux à partir d'autres littéraux, à l'aide de la fonction ARRAY : par exemple, `ARRAY(3, 5, 6)`, `ARRAY("Hello", 2.3, 9)`. Ces fonctions créent 3 tableaux avec les éléments entiers 3, 5 et 6 (premier exemple) et avec l'élément chaîne “ Hello”, l'élément double 2,3 et l'élément entier 9 (second exemple).

Fonctions : aucune représentation de littéral n'est disponible pour les fonctions. Les fonctions sont définies à l'aide du mot clé `FUNCTION` et accessibles via des ID de variables. Par exemple, `ASSIGN/Add2 = FUNCTION((X), X+2)` définit une fonction acceptant un argument et lui ajoute 2. La variable Add2 est attribuée à la fonction. La fonction peut être appelée à l'aide de la variable Add2 comme suit. `ASSIGN/Result = Add2(5)`. La valeur 7 est attribuée au résultat.

Remarque : les littéraux numériques sont considérés comme des nombres réels, à moins que l'opérateur ou la fonction n'implique l'utilisation d'entiers. Par exemple, l'expression 10 / 8 donne comme résultat 1,25 et non pas 1. A division discrète peut également se faire via les opérateurs de contrainte d'opérandes. L'expression INT(10) / INT(8) donne 1 comme résultat.

Références

Les références correspondent aux éléments de données d'autres objets dans une routine de mesure. Les références utilisent l'ID d'un objet figurant dans la routine de mesure, suivi d’un point et d’une extension relative à l'élément de données de cet objet.

Exemple : si CIRCLE1 est le nom d'un cercle mesuré dans la routine de mesure, `CIRCLE1.X` fait référence à la valeur mesurée du composant X du CIRCLE1. Toutes les références sont évaluées dans les coordonnées de pièce par rapport à l'alignement courant.

Références de type double

Les expressions de référence suivantes sont disponibles :

Extensions valides pour des références d'éléments de type double par exemple

Format : <ID élément>.<Extension> -> `CIRCLE1.X`

CIRCLE1.X Valeur X mesurée de CIRCLE1
CIRCLE1.Y Valeur Y mesurée de CIRCLE1
CIRCLE1.Z Valeur Z mesurée de CIRCLE1

CIRCLE1.TX Valeur X théorique (nominale) de CIRCLE1
CIRCLE1.TY Valeur Y théorique (nominale) de CIRCLE1
CIRCLE1.TZ Valeur Z théorique (nominale) de CIRCLE1

LINE1.SX Valeur X mesurée du point de départ de LINE1
LINE1.SY
LINE1.SZ

LINE1.TSX Valeur X théorique du point de départ de LINE1
LINE1.TSY
LINE1.TSZ

LINE1.EX Valeur X mesurée du point de fin de LINE1
LINE1.EY
LINE1.EZ

LINE1.TEX Valzur X théorique du point de fin de LINE1
LINE1.TEY
LINE1.TEZ

POINT.I Mesuré I Composant du vecteur pour POINT
POINT.J
POINT.K

POINT.TI Théorique I Composant du vecteur pour POINT
POINT.TJ
POINT.TK

FEAT1.TYP Le type de l'élément (comme cercle, logement, cône). Ceci peut servir à changer le type d'un élément générique (Assign/ Gen1.TYP = Feat1.TYP).

FEAT1.ALL fait référence à tous les composants de l'élément. Ceci s'avère utile quand vous copiez des informations dans un élément générique. (Assign/ Gen1.ALL = Feat1.ALL)

Vecteur de surface

EDGE.SURFI

EDGE.SURFJ

EDGE.SURFK

EDGE.TSURFI

EDGE.TSURFJ

EDGE.TSURFK

Vecteur d'angle

CIR.ANGI

CIR.ANGJ

CIR.ANGK

CIR.TANGI

CIR.TANGJ

CIR.TANGK

Rayon

CIRCLE1.R

CIRCLE1.TR

CIRCLE1.RAD

CIRCLE1.TRAD

CIRCLE1.RADIUS

CIRCLE1.PR – Rayon polaire

CIRCLE1.TPR – Rayon polaire théorique

CIRCLE1.TRADIUS (seuls les premiers caractères sont importants)

Diamètre

CIRCLE1.D

CIRCLE1.TD

CIRCLE1.DIAM

CIRCLE1.TDIAM

CIRCLE1.DIAMETER

CIRCLE1.TDIAMETER (seuls les premiers caractères sont importants)

Angle

CONE.A

CONE.TA

CONE.ANG

CONE.TANG

CONE.ANGLE

CONE.TANGLE

CONE.PA – Angle polaire

CONE.TPA – Angle polaire théorique (seuls les premiers caractères sont importants)

Longueur

LINE.L

LINE.TL

LINE.LEN

LINE.TLEN

LINE.LENGTH

LINE.TLENGTH (seuls les premiers caractères sont importants)

Hauteur

CYLINDER.PH – Hauteur polaire

CYLINDER.TPH – Hauteur polaire théorique

Rayon, Angle, Hauteur

POINT.RAH - Point avec rayon, angle et hauteur mesurés

POINT.TRAH – Point avec rayon, angle et hauteur théoriques

Zone

BLOB1.AREA - Renvoie la valeur de zone mesurée pour une élément Blob.

BLOB1.TAREA - Renvoie la valeur de zone théorique pour un élément Blob.

Par exemple, `ASSIGN/V1=BLOB1.AREA` renvoie la valeur de zone mesurée de l'élément BLOB1 et l'attribue à la variable V1.

Actuellement, seul l'élément Blob fonctionne avec ces extensions de zones. Pour des informations sur l'élément Blob, voir la rubrique "Blob Vision" dans la documentation "PC-DMIS Vision".

Extensions valides pour des références de dimensions de type double par exemple

Format : <Dimension ID>.<AXIS>.<Élément de dimension> -> `DIM1.X.NOM`

DIM1.X.NOM	La valeur nominale pour l'emplacement de DIM1 sur l'axe X
DIM1.X.MEAS	La valeur mesurée pour l'emplacement de DIM1 sur l'axe X
DIM1.X.MAX	La déviation maximale pour l'emplacement de DIM1 sur l'axe X
DIM1.X.MIN	La déviation minimale pour l'emplacement de DIM1 sur l'axe X
DIM1.X.PTOL	La valeur de tolérance positive pour l'emplacement de DIM1 sur l'axe X
DIM1.X.MTOL	La valeur de tolérance négative pour l'emplacement de DIM1 sur l'axe X
DIM1.X.DEV	La déviation de l'emplacement de DIM1 sur l'axe X
DIM1.X.OUTTOL	La valeur Hors tolérance de l'emplacement de DIM1 sur l'axe X
DIM1.Y.NOM	La valeur nominale pour l'emplacement de DIM1 sur l'axe Y
DIM1.Z.DEV	La déviation de l'emplacement de DIM1 sur l'axe Z
DIM3.PA.MEAS	La valeur mesurée pour l'emplacement de l'angle polaire de DIM3
DIM4.M.PTOL	La valeur de tolérance positive pour l'emplacement de DIM4 sur l'axe M
*DIM4.PTOL	La valeur de tolérance positive pour l'emplacement de DIM4 sur l'axe M (voir ci-dessous la remarque « Axes valides »).
DIM5.BTOL	La valeur de tolérance bonus où DIM5 est une localisation.

Axes valides :

X, Y, Z, D, R, A, T, V, L, PR, PA, M, PD, RS, RT, S, H, DD, DF, TP

Remarque : * les dimensions qui n'ont qu'un axe par définition (comme arrondi, concentricité, etc.) peuvent ignorer le qualificateur d'axe. Avec les qualificateurs d'axe, notez que tous ces types de dimensions (qui ne présentent qu'un seul axe) utilisent le qualificateur d'axe M, sauf les dimensions d'angles 2D et 3D qui ont recours au qualificateur d'axe A.

Extensions valides pour des références d'alignement de type double par exemple :

Format : <ID d'alignement>.<Origine ou axe d'alignement>.<Axe d'alignement ou composant initial> -> `A1.ORIGIN.X`

A1.ORIGIN.X	Composant X de l'origine mesurée de l'alignement A1
A2.ORIGIN.Y	Composant Y de l'origine mesurée de l'alignement A2
A1.ORIGIN.Z	Composant Z de l'origine mesurée de

	l'alignement A1
A1.XAXIS.I	Composant I de l'axe X mesuré de l'alignement A1
A1.YAXIS.J	Composant J de l'axe Y mesuré de l'alignement A1
A1.ZAXIS.K	Composant K de l'axe Z mesuré de l'alignement A1
A1.CORIGIN.X	Composant X de l'origine de l'alignement A1 basée sur les données théoriques (C = CAO)
A1.CXAXIS.J	Composant J de l'axe X de l'alignement A1 basé sur les données théoriques (C = CAO)

Références de type point

Les expressions de référence suivantes sont disponibles :

Extensions valides pour des références d'éléments de type point par exemple

Format : <ID élément>.<Extension> -> CIRCLE1.XYZ

CIRCLE1.XYZ	Barycentre mesuré de CERCLE1
CIRCLE1.TXYZ	Barycentre théorique de CERCLE1
LINE1.SXYZ	Point de départ mesuré de LIGNE1
LINE1.TSXYZ	Point de départ théorique de LIGNE1
LINE1.EXYZ	Point final mesuré de LIGNE1
LINE1.TEXYZ	Point final théorique de LIGNE1
CIRCLE1.IJK	Vecteur mesuré de CERCLE1
CIRCLE1.TIJK	Vecteur théorique de CERCLE1
EDGE.SURFIJK	Vecteur de surface mesuré de ARÊTE
EDGE.TSURFIJK	Vecteur de surface théorique de ARÊTE
AUTOCIR1.ANGIJK	Vecteur angulaire mesuré de AUTOCER1
AUTOCIR1.TANGIJK	Vecteur angulaire théorique de AUTOCER1

Extensions valides pour des références d'alignement de type point par exemple

Format : <ID alignement>.<Axe ou origine alignement> -> A1.XAXIS

A1.ORIGIN	Origine mesurée de l'alignement A1
A1.XAXIS	Axe X mesuré de l'alignement A1
A1.YAXIS	Axe Y mesuré de l'alignement A1
A1.ZAXIS	Axe Z mesuré de l'alignement A1
A1.CORIGIN	Origine théorique de l'alignement A1

A1.CXAXIS	Axe X théorique de l'alignement A1
A1.CYAXIS	Axe Y théorique de l'alignement A1
A1.CZAXIS	Axe Z théorique de l'alignement A1

Références de type chaîne

Les références à des commentaires sont les seuls objets de type chaîne. Seuls les commentaires d'ENTRÉE ou de type OUI/NON peuvent être référencés par l'intermédiaire de références. L'ID de ces commentaires sert à les désigner.

Format : <ID commentaire>.INPUT -> `C1.INPUT`

C1.INPUT - Valeur saisie par l'opérateur concernant le commentaire C1

Les commentaires de type OUI/NON définissent la saisie dans la chaîne en fonction du langage en cours de PC-DMIS. Dans la version anglaise de PC-DMIS, si l'opérateur clique sur le bouton Oui, la chaîne est définie sur « YES » ; s'il clique sur le bouton Non, elle est définie sur « NO ». La comparaison des chaînes « YES » et « NO » tient compte des majuscules. Une comparaison avec « yes » ou « no » échoue forcément même si la saisie du commentaire est en majuscules (« YES » ou « NO »).

Variables

Les variables peuvent appartenir à l'un des sept types d'opérandes : entier, réel, chaîne, point, pointeur d'élément , tableau ou fonction. Les variables sont créées et reçoivent leurs valeurs et type via l'instruction `ASSIGN`. L'ID de variable peut être n'importe quelle chaîne alphanumérique, à condition qu'elle ne commence pas par un chiffre. Les traits de soulignement sont autorisés, à condition qu'ils ne représentent pas le premier caractère de l'ID.

Les valeurs des variables sont enregistrées entre les exécutions. Cela implique que lorsque l'exécution de la routine de mesure s'arrête et reprend, les valeurs affectées aux variables ne changent pas à la reprise de l'exécution.

Remarque : si la fenêtre de modification est active, PC-DMIS indique la valeur courante de la variable chaque fois que le curseur est placé dans cette zone. Pendant l'exécution, les valeurs des variables varient selon le flux des opérations. Placez le pointeur de la souris sur la variable dont vous voulez connaître la valeur courante.

```
ASSIGN/ V1 = 2.2+2
```

La variable V1 est un nombre réel avec la valeur 4,2

```
ASSIGN/ VAR1 = CIRCLE1.X
```

La variable VAR1 est un nombre réel dont la valeur correspond à la valeur mesurée de CIRCLE1.X au moment de l'affectation.

```
ASSIGN/ MYVAR = LINE1.XYZ
```
La variable MYVAR est un point de même valeur que le barycentre mesuré de LINE1 au moment de l'affectation.

```
ASSIGN/ SVAR = "Hello World"
```
La variable SVAR est une chaîne dont la valeur correspond à 'Hello World".

Dans ces exemples, des valeurs sont affectées aux différentes variables. Dès qu'une variable a une valeur, elle peut servir d'opérande dans n'importe quel champ d'expression.

Exemple de valeur V1 utilisée dans une zone numérique :
```
ASSIGN/V1 = 1/3
PREHIT / V1
```

V1 est la valeur de prépalpage de la commande de prépalpage.

Remarque : les expressions pouvant être utilisées dans la plupart des champs à éditer, l'expression suivante est tout aussi acceptable et a le même effet : PREHIT / 1/3.

Les références aux composants des variables de type point peuvent se faire individuellement grâce à la notation avec extension utilisée pour les références.

```
ASSIGN/ V1 = MPOINT(3, 4, 5)
```
V1 est de type point avec la valeur 3, 4, 5.

```
ASSIGN/ XVAR = V1.X
```
XVAR est de type double avec la valeur 3.

```
ASSIGN/ YVAR = V1.Y
```
YVAR est de type double avec la valeur 4.

```
ASSIGN/ IVAR = V1.I
```
IVAR est de type double avec la valeur 3.

```
ASSIGN/ REDUNVAR = V1.XYZ
```
REDUNVAR est de type point avec la valeur 3, 4, 5.

Les extensions suivantes sont équivalentes. Elles servent à clarifier la signification d'une expression dans une routine de mesure.

Étant donné que V1 est de type point :

V1.X équivaut à V1.I
V1.Y équivaut à V1.J
V1.Z équivaut à V1.K

V1.XYZ équivaut à V1.IJK et à V1 sans extension.

Si une variable de type chaîne a une valeur égale à l'ID d'un élément, d'une dimension ou d'un alignement, elle peut servir d'objet de référence.

Exemple :

```
ASSIGN/ V1 = "CIRCLE1"
```

Les opérandes suivants sont possibles et valides, à condition qu'un élément appelé CIRCLE1 existe.

V1.X - Valeur mesurée de CIRCLE1
V1.TX - Valeur X théorique de CIRCLE1
V1.Diameter - Diamètre mesuré de CIRCLE1
V1.Radius - Rayon mesuré de CIRCLE1

Ce type d'adressage indirect utilisable avec les variables de chaîne n'est disponible qu'à un seul niveau d'adressage indirect. L'expression suivante est incorrecte :

```
ASSIGN/ V1 = "CIRCLE1"
ASSIGN/ V2 = "V1"
V2.X - Donne 0 comme résultat au lieu de la valeur mesurée courante de CERCLE1.X.
```

Remarque : la référence V2.X n'est *pas* signalée comme erronée (texte en rouge) bien que l'expression précédente la définisse comme chaîne. La raison est simple : le flux des opérations de la routine de mesure reste inconnu jusqu'au moment de l'exécution.

Toutefois, si vous utilisez des accolades, ce qui suit s'applique :

```
ASSIGN/ V1 = {CIRCLE1}
ASSIGN/ V2 = {V1}
V2.X - Donne la valeur de CIRCLE1.X.
```

Prenez l'exemple suivant :

```
ASSIGN/ V1 = "CIRCLE1"
ASSIGN/ V2 = "V1"
IF / CIRCLE1.X > CIRCLE1.TX, GOTO, L2
L1 = LABEL /
ASSIGN/ V3 = V2.X
GOTO / LABEL, L3
L2 = LABEL /
ASSIGN/ V2 = MPOINT( 2, 5, 7)
GOTO / LABEL, L1
L3 = LABEL /
```

Pendant l'exécution de la routine, si la valeur CIRCLE1.X est supérieure à CIRCLE1.TX, l'expression V2.X est valide et est évaluée à 2. Sinon, l'expression V2.X est évaluée à 0 car la valeur V2 avec ASSIGN pour V3 est la chaîne "V1". Il appartient au programmeur de s'assurer que les expressions donnent les résultats attendus dans ces cas.

Remarque supplémentaire concernant l'instruction Assign : vous pouvez utiliser presque toutes les références d'éléments qui se trouvent à gauche de l'instruction d'affectation pour attribuer une valeur au composant de donnée mesuré ou théorique d'un élément. Seules exceptions à la règle : les composants I, J, K uniques de vecteurs. L'affectation doit porter sur

tout le vecteur à la fois et l'expression utilisée doit donner un point comme résultat. Les données vectorielles sont normalisées à mesure de leur entrée dans leurs composants d'élément.

Exemple :

```
ASSIGN/ CIRCLE1.I = 2 - illégal
ASSIGN/ CIRCLE1.IJK = MPOINT(2, 0, 0)-légal (vecteur normalisé à 1, 0, 0)
```

Pour en savoir plus sur l'utilisation de variables à l'intérieur des dimensions, voir la rubrique "Cotation de variables" au chapitre "Cotation d'éléments".

Structures

Vous pouvez utiliser une zone de variable appelée *structures* afin de placer des extensions sur une variable et identifier un sous-élément de cette variable. Prenez l'exemple suivant :

Exemple :

```
ASSIGN/V1.HEIGHT = 6
ASSIGN/V1.WIDTH = 4.3
ASSIGN/V1.MODE = "CIRCULAR"
ASSIGN/V1.POINT = MPOINT(100.3, 37.5, 63.1)
```

Dans cet exemple :

- `V1` est la structure.
- `HEIGHT`, `WIDTH`, `MODE` et `POINT` sont des sous-éléments de la structure.

Règles de structures

- Comme pour les variables, vous n'êtes pas tenu de déclarer les structures.
- Les sous-éléments d'une structure peuvent être de l'un des types suivants de variable :
    - Entier
    - Double
    - Point
    - Pointeur d'élément
    - Fonction
    - Tableau
    - Structure

Par exemple, il est possible d'avoir des éléments de structure de type tableaux et des éléments de tableau de type structures. L'exemple d'expressions suivant est ainsi valide :

Exemple

```
ASSIGN/CAR.LEFTSIDE.DOOR[2].QUADRANT[3].JOINT[5].HIT [4] = MPOINT(558.89, 910.12, 42.45)

COMMENT/OPER,"Current Z Position: " + CAR.LEFTSIDE.DOOR[2].QUADRANT[3].JOINT[5].HIT[4].Z

ASSIGN/CURRENTJOINT = LEFTSIDE.DOOR[2].QUADRANT[3].JOINT[5]
```

```
COMMENT/OPER,"Next Hit: " + CURRENTJOINT.HIT[4]
```

Structures avec des variables de type point

Si une variable est de type point, l'utilisateur peut tout de même utiliser les extensions .X, .Y, .Z, .I, .J, and .K pour arriver à des éléments individuels du point. L'utilisateur peut également utiliser l'une des extensions de cet exemple dans leur structures sans devoir les utiliser comme éléments de point.

Exemple :

```
ASSIGN/V1.X = "Une chaîne"
ASSIGN/V1.Y = ARRAY(1,3,5,9,7)
ASSIGN/V1.Z = MPOINT(3,5,7)


COMMENT/REPT,V1.X           La sortie est « Une
                            chaîne »

COMMENT/REPT,V1.Y[2]        La sortie est 3, le
                            deuxième élément du
                            tableau.

COMMENT/REPT,V1.Z.Y         La sortie est 5, la
                            valeur Y de MPOINT.
```

En combinant les structures avec la capacité des fonctions du langage d'expression de PC-DMIS, il est possible d'obtenir des références de structure dynamique, telles que les suivantes :

(exemple - suite)

```
ASSIGN/DYNAMICSTRUCT = FUNCTION((X,Y), X.Y)
C1 =COMMENT/INPUT,Please enter in item
ASSIGN/TESTSTR = C1.INPUT
ASSIGN/FRONT = LEFT(TESTSTR, INDEX(TESTSTR, ".")-1)
ASSIGN/BACK = MID(TESTSTR, INDEX(TESTSTR, "."))
ASSIGN/RESULT = DYNAMICSTRUCT(FRONT, BACK)
```

Cette partie de l'exemple vous demande d'entrer une référence de variable, de fractionner la référence au premier '.', puis d'affecter à `RESULT` une valeur égale à cette référence à l'aide de la fonction `DYNAMICSTRUCT`.

Si vous avez saisi `V1.Y[4]` pour la variable C1.INPUT, la valeur 9 (quatrième élément du tableau) est assignée à V1.Y).

L'évaluation du temps d'apprentissage des expressions a été améliorée pour afficher tous les éléments d'une structure ou d'un tableau de manière précise.

Pointeurs

Les pointeurs sont également appelés « pointeurs d'élément ». Pour plus d'informations, voir l'entrée « Pointeurs d'élément »dans le glossaire de l'aide en ligne.

Les pointeurs offrent un moyen de référencer un élément par l'intermédiaire d'une variable ou de transfert d'objets à l'aide de la sous-commande d'appel. Ils ressemblent à l'adressage indirect par noms de chaînes. Leur avantage est lié à l'utilisation de sous-programmes. Contrairement aux chaînes, les pointeurs transférés en tant qu'arguments de sous-programme autorisent la modification directe de l'objet désigné par le sous-programme. Ils ne s'utilisent pas dans les expressions complexes. Tout pointeur utilisé dans une expression complexe donne zéro comme résultat.

Prenez les exemples suivants :

Exemple d'utilisation de pointeur :

- `ASSIGN/ V1 = {CIR1}`

  V1 est maintenant un pointeur pointant vers CER1.

- `ASSIGN/ DIST = DOUBLE(V1.XYZ)`

  Dist = distance de CER1 depuis l'origine.

Vous pouvez également placer une expression entre accolades de manière à obtenir un pointeur d'élément. Les exemples suivants vous permettent désormais de placer le pointeur sur l'élément CER1 de manière légale :

- ```
  ASSIGN/FEATCOUNT = 1
  ASSIGN/V1={"CIR" + FEATCOUNT}
  ```

  Affecte l'expression « CER1 » à V1.

- ```
  ASSIGN/V2="CIR1"
  ASSIGN/V3={V2}
  ```

  Affecte l'expression « CIR1 » à partir de la variable V2 vers la variable V3.

- ```
  C1=COMMENT/INPUT, Veuillez saisir un nom d'élément.
  ASSIGN/V4={C1.INPUT}
  ```

  Cette opération prend le nom d'élément de C1.INPUT et le place dans la variable V4.

Exemple de sous-programme :

Dans la routine d'appel :

```
CS1 = APPELSP/SP.PRG,CHANGEX, {CER1}
```

Dans le sous-programme :
```

```
GEN1 = ÉLÉMENT/GÉNÉRIQUE
SOUS-PROGRAMME/CHANGEX, ARG1 = {GÉN1}
```

(Une fois entré, CER1 prend la place de GÉN1)

```
ARG1.X = 5
```

(Définit la valeur X mesurée de CER1 à 5)

```
FIN/SOUS-PROGRAMME
```

Exemple d'expression complexe :

```
AFFECT/ V1 = {CER1} + 2
```

{CER1} a zéro pour valeur et l'expression entière a 2 pour valeur.

Tableaux

Trois types de tableaux sont disponibles : d'éléments, de palpages et de variables.

Important : même si des tableaux multidimensionnels sont affichés comme tels dans le logiciel, vous pouvez seulement les utiliser comme tableaux d'une seule dimension tant que vous ne placez pas une commande ARRAY INDICES avant (voir la rubrique "Objet d'index de tableau").

Tableaux d'éléments :

Il suffit qu'un élément soit mesuré plus d'une fois pendant l'exécution de la routine pour qu'un tableau soit automatiquement créé. Le nombre d'entrées dans le tableau d'éléments est égal au nombre de fois où l'élément a été exécuté.

Exemple : si un objet de cercle mesuré se trouve dans une boucle conditionnelle exécutée cinq fois, alors un tableau de cinq cercles mesurés est créé. Si l'ID du cercle mesuré est "Circle1", alors une expression de tableau peut être utilisée pour accéder aux instances individuelles de cet objet. Les crochets servent à indiquer l'instance désirée

```
Assign/V1 = Circle1[3].X
```

La valeur x mesurée de la troisième instance de Circle1 est affectée à V1.

Remarque : quand il existe un tableau pour un élément donné mais que la notation de tableau n'est pas utilisée pour référencer cet élément, l'instance la plus récente est utilisée. Pour reprendre l'exemple précédent, la référence Circle1.X serait identique à Circle1[5].X puisque la cinquième instance est en fait l'instance la plus récente de l'objet.

Les expressions peuvent s'utiliser entre les crochets d'une expression de tableau. Circle1[3].X et Circle1[2+1].X sont donc des expressions équivalentes. L'exemple suivant applique une boucle pour imprimer le barycentre mesuré des cinq cercles tirés des exemples précédents.

Exemple :

```
Assign/V1 = 1
While/V1 < 6
Comment/Rept, "Barycentre de l'instance #" + V1 + " de Circle1: " + Circle1[V1].XYZ
Assign/V1 = V1 + 1
End/While
```

Résultat possible à partir de l'exemple précédent :

Barycentre de l'instance 1 de Cercle1 : 3,4, 2,6, 1,43

Barycentre de l'instance 2 de Cercle1 : 4,4, 3,6, 2,43

Barycentre de l'instance 3 de Cercle1 : 5,4, 4,6, 3,43

Barycentre de l'instance 4 de Cercle1 : 6,4, 5,6, 4,43

Barycentre de l'instance 5 de Circle1 : 7,5, 6,6, 5,43

Des tableaux existent aussi pour les dimensions et les alignements qui ont été exécutés plusieurs fois pendant le même tour d'exécution. Ainsi, Dim1[2].Nom et Align1[4].Origin sont donc disponibles si la dimension "Dim1" a été exécutée au moins deux fois et l'alignement "Align1" au moins quatre.

Si une référence de tableau d'éléments est hors limite (l'utilisateur demande par exemple Circle1[2,5] ou > Circle1["Hello, World"]), l'élément limite supérieur ou inférieur est renvoyé. Si Circle1 a 3 instances, Circle1[4] et au-dessus retournent Circle1[3] tandis que Circle[0] et au-dessous retournent Circle1[1]. Toutes les expressions entre crochets prennent obligatoirement la valeur d'un entier : ainsi 2,5 devient 2 et "Hello, World" devient 0.

Objet d’index de tableau

Par défaut, les tableaux d'éléments sont toujours unidimensionnels. S'il est plus simple de traiter un tableau d'éléments sous sa forme multidimensionnelle, il convient d'utiliser l'objet d'index de tableau.

L'objet d'index de tableau vous permet de spécifier des limites supérieures et inférieures pour les tableaux multidimensionnels.

- Lorsque vous spécifiez les limites supérieures et inférieures de la première dimension, un tableau bidimensionnel est créé, sa première dimension étant limitée et sa seconde illimitée.
- Lorsque vous spécifiez les limites supérieures et inférieures des deux premières dimensions d'un tableau, un tableau tridimensionnel est créé. La dernière dimension est toujours illimitée.

Exemple :

L'élément F1 est situé dans une boucle PENDANT intégrée. La boucle PENDANT interne est exécutée cinq fois et la boucle PENDANT externe trois fois. Au terme de l'exécution, F1 a été exécuté 15 fois et 15 instances de F1 ont été créées.

Prenez l'exemple de segment de routine de mesure suivant :

```
INDEX TABLEAU/1..5,..
ASSIGN/V1 = 1
WHILE/V1<=3
ASSIGN/V2 = 1
WHILE/V2<=5
F1=FEAT/POINT,RECT
THEO/V2,V1,0,0,0,1
ACTL/1,1,0,0,0,1
MEAS/POINT,1
HIT/BASIC,V2,V1,0,0,0,1,1,1,0
ENDMEAS/
ASSIGN/V2 = V2+1
COMMENT/REPT,"Emplacement de F1[" + V2 + "," + V1 + "] :" + F1[V2,V1].XYZ
END_WHILE/
ASSIGN/V1 = V1+1
END_WHILE/
```

Ce segment de code crée une grille 3 X 5 de 15 points mesurés.

La commande d'index de tableau limite la première dimension du tableau d'éléments à une valeur comprise entre 1 et 5 inclus. Ainsi, au lieu de voir F1[1] – F1[15] dans le rapport d'inspection, les objets apparaissent sous la forme F1[1, 1] – F1[5, 3], ce qui est plus cohérent avec la présentation des éléments. Vous remarquez que le commentaire renvoie également au tableau d'éléments avec une syntaxe de tableau bidimensionnel.

Pour insérer un objet d'index de tableau dans une routine de mesure :

1. À l'aide du clavier, entrez "**Tableau**" sur une ligne vierge de la fenêtre de modification.
2. Appuyez sur la touche Tab de votre clavier.

Remarque : si la case **Afficher crochets pour tableaux d'éléments** est décochée, l'élément n'apparaît pas avec le nom entre crochets. Voir "Afficher crochets pour tableaux d'éléments" au chapitre "Définition des préférences".

Tableaux de palpages

Les palpages d'un élément donné sont disponibles sous forme de tableaux et sont accessibles par le biais d'expressions dont la syntaxe est de type <ID élém>.Hit[<Expression tableau>].<Extension> ou <ID élém>.RawHit[<Expression tableau>].<Extension>. Si la fonction de compensation est activée, Hit retourne les données compensées du palpeur. RawHit renvoie toujours des données non compensées. Les extensions valides sont X, Y, Z, I, J, K, TX, TY, TZ, TI, TJ, TK, XYZ, TXYZ, IJK et TIJK.

`Circle1.Hit[1].XYZ`
Barycentre mesuré (avec compensation palpeur) du palpage 1 de "Circle1".

`Circle1.Hit[2].IJK`
Vecteur mesuré du palpage 2 de "Circle1.

Les données de palpage sont disponibles pour tous les objets palpés, que les palpages réels s'affichent ou non dans la fenêtre Édition. Vous pouvez ainsi obtenir des palpages pour des scans et des éléments automatiques.

Les rubriques suivantes décrivent d'autres fonctions de tableau servant à rechercher les points minimum et maximum dans un scan :

Affectation d'une gamme de palpages à un tableau

Vous pouvez également affecter une gamme de palpages à un tableau à l'aide de cette syntaxe :

<ID élém>.<Type palpage>[<Num début>..<Num fin>].<Extension>

où

<ID élém> est le nom de l'élément.

<Hittype> peut correspondre à "HIT" pour les données compensées ou à "RAWHIT" pour les données non compensées. Si la compensation du palpeur est désactivée, les valeurs de retour ne sont jamais compensées.

<Num début> est une expression identifiant la première valeur d'index de la gamme de palpages.

<Num fin> est une expression identifiant la seconde valeur d'index de la gamme de palpages.

<Extension> identifie le type de données. Les extensions possibles incluent ces types de données mesurées ou théoriques :

- X – Valeurs mesurées X des palpages
- Y – Valeurs mesurées Y des palpages
- Z – Valeurs mesurées Z des palpages
- XYZ – Valeurs mesurées XYZ des palpages
- I – Valeurs mesurées I des palpages
- J – Valeurs mesurées J des palpages
- K – Valeurs mesurées K des palpages
- IJK – Valeurs mesurées IJK des palpages

- TX – Valeurs théoriques X des palpages
- TY – Valeurs théoriques Y des palpages
- TZ – Valeurs théoriques Z des palpages
- TXYZ – Valeurs théoriques XYZ des palpages
- TI – Valeurs théoriques I des palpages
- TJ – Valeurs théoriques J des palpages

- TK – Valeurs théoriques K des palpages
- TIJK – Valeurs théoriques IJK des palpages

Par exemple,

```
ASSIGN/V1 = SCAN1.HIT[1..10].X
```

V1 est affecté à un tableau de 10 valeurs X mesurées à partir des 10 premiers palpages de SCAN1.

```
ASSIGN/V2 = SCAN1.HIT[1..SCAN1.NUMHITS].XYZ
```

V2 est affecté à un tableau de points à partir de chaque barycentre des palpages dans le scanning.

Tri de tableaux

PC-DMIS vous permet de trier des tableaux dans l'ordre croissant ou décroissant. Les deux expressions suivantes prennent un tableau et le renvoient trié :

Pour trier dans l'ordre *croissant* utilisez :

```
SORTUP(<tableau>)
```

Pour trier dans l'ordre *décroissant* utilisez :

```
SORTDOWN(<tableau>)
```

Par exemple :

```
ASSIGN/V1 = ARRAY(5,8,3,9,2,6,1,7)
```

Le tableau "5,8,3,9,2,6,1,7" est affecté à V1.

```
ASSIGN/V2 = SORTUP(V1)
```

V2 conserve les valeurs du tableau triées dans l'ordre croissant : "1,2,3,5,6,7,8,9"

```
ASSIGN/V3 = SORTDOWN(V1)
```

V3 conserve les valeurs du tableau triées dans l'ordre décroissant : "9,8,7,6,5,3,2,1"

Renvoi des valeurs d’index les plus élevées ou les plus faibles à partir d’un tableau

Vous pouvez insérer un tableau à une fonction et renvoyer le numéro d’index de l’élément possédant la valeur la plus élevée ou la plus fable à l’aide de ces fonctions :

Pour renvoyer la valeur d'index de l'élément possédant la valeur *la plus élevée*, utilisez :

```
MAXINDEX(<tableau>)
```

Pour renvoyer la valeur d'index de l'élément possédant la valeur *la moins élevée*, utilisez :

```
MININDEX(<tableau>)
```

Par exemple :

```
ASSIGN/V1 = ARRAY(5,8,3,9,2,6,1,7)
```
Le tableau "5,8,3,9,2,6,1,7" est affecté à V1.

```
ASSIGN/V2 = MAXINDEX(V1)
```
V2 conserve la valeur d'index 4 du tableau. La valeur réelle de cet élément de tableau est 9.

```
ASSIGN/V3 = MININDEX(V1)
```
V3 conserve la valeur d'index 7 du tableau. La valeur réelle de cet élément de tableau est 1.

Vous pouvez ensuite utiliser les valeurs d'index renvoyées pour obtenir la valeur réelle de l'élément.

Renvoi de valeurs d'index triées à partir d'un tableau

Vous pouvez insérer un tableau dans une fonction, en trier les valeurs dans l'ordre croissant ou décroissant et renvoyer ensuite les valeurs d'index à l'aide de ces fonctions :

Pour renvoyer les positions d'index du tableau dans l'ordre des valeurs triées de la plus élevée à la plus faible, utilisez :
```
MAXINDICES(<tableau>)
```

Pour renvoyer les positions d'index du tableau dans l'ordre des valeurs triées de la plus faible à la plus élevée, utilisez :
```
MININDICES(<tableau>)
```

Par exemple :

ASSIGN/V1 = ARRAY(4,8,2,9,5,7)
Le tableau "4,8,2,9,5,7" est affecté à V1.

ASSIGN/V2 = MAXINDICES(V1)
V2 conserve un tableau avec ces valeurs : "4,2,6,5,1,3".

ASSIGN/V3 = MININDICES(V1)
V3 conserve un tableau avec ces valeurs : "3,1,5,6,2,4".

Exemple d’utilisation de fonctions de tableau pour rechercher les points minimum et maximum dans un scanning

L'objet principal des fonctions de tableau présentées ci-dessus consiste à faciliter la recherche des points minimum et maximum dans un scan.

Pour coter le point à partir de SCAN1 possédant la valeur X mesurée la plus élevée, vous pouvez utiliser cette expression :

```
ASSIGN/MAXPTINDEX = MAXINDEX(SCAN1.HIT[1..SCAN1.NUMHITS].X)
D1 = LOCATION OF FEATURE SCAN1.HIT[MAXPTINDEX]
```

Pour rechercher les trois points les plus élevés dans l'axe Z de SCAN2, vous pouvez utiliser cette expression :

```
ASSIGN/MI = MAXINDICES(SCAN2.HIT[1..SCAN2.NUMHITS].Z)
ASSIGN/THREEPOINTS = ARRAY(SCAN2.HIT[MI[1]].XYZ, SCAN2.HIT(MI[2]].XYZ,
SCAN2.HIT[MI[3]].XYZ)
```

Tableaux de variables

Vous n'êtes pas tenu de déclarer les tableaux de variables. Les tableaux de variables sont générés par l'instruction d'affectation quand l'expression à droite de celle-ci donne comme résultat un tableau ou quand celle à sa gauche fait référence à un élément d'un tableau de variables.

`Assign/V1 = Array(3, 4, 5, 6, 7)`
Crée un tableau de 5 éléments et l'affecte à V1.

`Assign/V2 = V1[3]`
Affecte à V2 la valeur du troisième élément dans le tableau V1: 5.

`Assign/V1[4] = 23`
Affecte le 4e élément du tableau V1 à 23.

Les tableaux sont créés et attribués de façon dynamique. Vous pouvez donc créer un tableau en utilisant sa référence à gauche d'une instruction d'affectation.

`Assign/V3[5] = 8`
Crée de façon dynamique un tableau avec le 5e élément égal à 8.

Si vous faites référence à un composant de tableau qui n'a jamais reçu de valeur, l'expression donne 0 comme résultat.

`Assign/V3[5] = 8`

`Assign/V4 = V3[5]`
V4 est défini égal à 8.

`Assign/V5 = V3[6]`
Si le sixième élément de V3 n'a pas été défini, V5 est égal à 0.

Comme d'autres types de tableaux, les expressions peuvent être utilisées entre crochets.

```
Assign/V3[5] = 8
```

```
Assign/V4 = V3[2+3]
```

V4 est défini égal à 8.

Les tableaux de variables peuvent avoir plusieurs dimensions.

```
Assign/V6 = Array( Array(4, 7, 2), Array(9, 2, 6))
```

V6 est défini à 2 par un tableau 3D où V6[1, 1] est égale à 4, V6[1, 2] à 7, V6[1, 3] à 2, V6[2, 1] à 9, V6[2,2] à 2 et V6[2,3] à 6.

```
Assign/V7 = V6[2,1]
```

V7 est défini à 9.

Les tableaux de variables peuvent avoir des index négatifs :

```
Assign/V8[-3] = 5
```

Le 3e index du tableau V8 est défini à 5.

L'affectation de tableau remplace les valeurs précédentes :

```
Assign/V8 = "Hello"
```

La variable V8 est égale à la chaîne "Hello"

```
Assign/V8[2] = 5
```

V8 n'est plus de type string, mais de type array, et son second élément a la valeur 5.

```
Assign/V8 = 9
```

V8 n'est plus un tableau mais un entier avec la valeur 9.

Les tableaux peuvent regrouper plusieurs types :

```
Assign/V9 = Array("Hello", 3, 2.9, {FEAT1})
```

Crée un tableau V9 avec 4 éléments. Le premier est une chaîne, le deuxième un entier, le troisième un nombre réel et le quatrième un pointeur vers l'élément FEAT1.

Vous pouvez agrandir les tableaux pour inclure plus d'éléments :

```
ASSIGN/V10=ARRAY(3,1,5)
```

```
ASSIGN/V10[LEN(V1)+1]=7
```

La première instruction crée un tableau initial V10 avec 3 éléments (3,1 et 5). La seconde instruction agrandit le tableau dans V10 d'un élément et attribue à l'élément final la valeur 7.

Opérateurs pour les expressions

Les opérateurs de base suivants sont disponibles dans PC-DMIS :

+ Addition : *<Expression>* + *<Expression>*
Fait la somme des deux expressions. S'il s'agit de chaînes, elles sont concaténées.

- Soustraction : *<Expression>* - *<Expression>*
Soustrait la seconde expression de la première.

***** Multiplication : *<Expression>* * *<Expression>*
Multiplie les deux expressions.

/ Division : *<Expression>* / *<Expression>*
Divise la première expression par la seconde.

^ Exposant : *<Expression>* ^ *<Expression>*
La première expression est élevée à la puissance de la seconde.

% Modulo : <Expression> % <Expression>
Retourne le reste d'une expression divisée par l'autre.

- Inverse additif : -*<Expression>*
Renvoie l'inverse additif de l'expression.

! Non logique : *!<Expression>*
Renvoie le non logique de l'expression.

== Égal à : *<Expression>* == *<Expression>*
Donne 1 comme résultat si les expressions sont égales. Dans les autres cas, le résultat est 0. Deux signes d'égalité permettent de distinguer cette fonction de l'opérateur = dans l'instruction d'affectation.

<> Non égal à : *<Expression>* <> *<Expression>*
Donne 1 comme résultat si les expressions ne sont pas égales. Dans les autres cas, le résultat est 0.

> Supérieur à *:* *<Expression>* > *<Expression>*
Donne 1 comme résultat si la première expression est supérieure à la seconde. Dans les autres cas, le résultat est 0.

>= Supérieur ou égal à : *<Expression>* >= *<Expression>*
Donne 1 comme résultat si la première expression est supérieure ou égale à la seconde. Dans les autres cas, le résultat est 0.

< Inférieur à : *<Expression>* < *<Expression>*
Donne 1 comme résultat si la première expression est inférieure à la seconde. Dans les autres cas, le résultat est 0.

<= Inférieur ou égal à : *<Expression>* <= *<Expression>*
Donne 1 comme résultat si la première expression est inférieure ou égale à la seconde. Dans les autres cas, le résultat est 0.

AND And logique : *<Expression> AND <Expression>*
Donne 1 comme résultat si les deux expressions ne sont pas évaluées à 0. Dans les autres cas, le résultat est 0.

OR Or logique : *<Expression> OR <Expression>* Donne 1 comme résultat si les deux expressions ne sont pas évaluées à 0. Dans les autres cas, le résultat est 0.

() Parenthèses : *(<Expression>)*
Donne priorité dans l'ordre d'évaluation à l'expression entre parenthèses.

Priorité

L'évaluation des expressions se fait dans l'ordre décroissant de priorité suivant :

Première priorité

- Opérandes
- (moins unaire), !, (), fonctions (comme ABS, COS, STR, LEN, CROSS, etc.)
- ^
- *, /, %
- +, -
- ==, <>, <, <=, >, >=
- ET
- OU

Dernière priorité

Fonctions

Les fonctions sont des expressions spécifiques de PC-DMIS ou des expressions définies par l'utilisateur qui acceptent normalement des paramètres et renvoient des résultats. Les paramètres sont remplacés dans l'expression avant l'évaluation de celle-ci.

Liste de fonctions

La liste alphabétique suivante contient toutes les fonctions disponibles pour le langage d'expression de PC-DMIS.

- ABS (mathématique)
- ACOS (mathématique)
- ANGLEBETWEEN (point)
- ARCSEGMENTENDINDEX (divers)
- ARCSEGMENTSTARTINDEX (divers)
- ARRAY (tableau)
- ASIN (mathématique)
- ATAN (mathématique)
- CHR (chaîne)
- COS (mathématique)
- CROSS (point)
- DEG2RAD (mathématique)
- DELTA (point)
- DIST2D (pointeur)
- DIST3D (pointeur)
- DOT (point)
- ELEMENT (chaîne)
- EOF (divers)
- EOL (divers)
- EQUAL (tableau)
- EQUAL (chaîne)
- EXP (mathématique)
- FORMAT (chaîne)
- FUNCTION (fonction)
- GETCOMMAND (pointeur)
- GETSETTING (chaîne)
- GETTEXT (chaîne)
- IF (divers)
- INDEX (chaîne)
- ISIOCHANNELSET (divers)
- LEFT (chaîne)
- LEN (tableau)
- LEN (pointeur)
- LEN (chaîne)
- LINESEGMENTENDINDEX (divers)
- LINESEGMENTSTARTINDEX (divers)
- LN (mathématique)
- LOG (mathématique)
- LOWERCASE (chaîne)
- MAX (tableau)
- MID (chaîne)
- MIN (tableau)
- MPOINT (point)
- ORD (chaîne)
- PCDMISAPPLICATIONPATH (chaîne)
- PCDMISUSERHIDDENDATAPATH (chaîne)

- PCDMISUSERVISIBLEDATAPATH (chaîne)
- PCDMISSYSTEMHIDDENDATAPATH (chaîne)
- PCDMISSYSTEMVISIBLEDATAPATH (chaîne)
- PCDMISSYSTEMREPORTINGPATH (chaîne)
- PROBEDATA (divers)
- QUALTOOLDATA (divers)
- RAD2DEG (mathématique)
- RIGHT (chaîne)
- ROUND (mathématique)
- SIN (mathématique)
- SQRT (mathématique)
- SYSTEMDATE (chaîne)
- SYSTEMTIME (chaîne)
- SYSTIME (chaîne)
- TAN (mathématique)
- TUTORELEMENT (divers)
- UNIT (point)
- UPPERCASE (chaîne)

Fonctions de chaîne

Les fonctions suivantes sont utilisées avec des chaînes de texte :

CHR

CHR
Conversion de caractère : *CHR(<entier>)*

Renvoie une chaîne composée du caractère correspondant à la valeur décimale ASCII.

ELEMENT

ELEMENT
Emplacement de sous-chaîne délimité : *ELEMENT(<Entier>, <Chaîne1>, <Chaîne2>)*

Renvoie la énième sous-chaîne (composant) de la chaîne2 en utilisant la chaîne1 comme texte de séparation qui divise le composant dans la chaîne2. Exemple : si la chaîne2 est "6, 12, 8, 4, 5" et la chaîne1 ",", vous pouvez récupérer séparément avec la commande FEATURE les 5 composants (6, 12, 8, 4 et 5).

EQUAL

EQUAL
Comparaison de chaînes respectant la casse : *EQUAL(<Chaîne>, <Chaîne>)*

Compare deux chaînes sans tenir compte des majuscules/minuscules pour voir si elles sont identiques. Renvoie un entier défini sur 1 si les chaînes sont identiques, ou 0 dans le cas contraire.

FORMAT

FORMAT
Format : FORMAT(<Chaîne>,<Entier,double ou point>)

Cette fonction prend deux expressions et renvoie une chaîne formatée, semblable à la fonction *sprintf* dans C++.

- La première expression doit être de type *chaîne* et contenir un ou trois spécificateurs de format. Si elle est d'un autre type, l'évaluateur d'expression tente de la forcer au type chaîne. La chaîne doit contenir *un* spécificateur de format si la seconde expression est de type entier ou double et *trois* spécificateurs de format (voir paragraphes ci-dessous) si cette expression est de type point.
- La seconde expression doit être de type *entier*, *double* ou *point*. Si un autre type est employé, la valeur de l'expression est 0.

Spécificateur de format pour la fonction FORMAT :

Le spécificateur de format doit obéir à la même syntaxe que celui employé dans la fonction *sprintf* utilisée en langage de programmation C++.

Un spécificateur de format se compose de champs requis et facultatifs, avec la syntaxe suivante :

%[flags] [width] [.precision] type

Chaque zone contient un caractère ou un chiffre désignant une option de format donnée. Le spécificateur de format le plus simple utilise uniquement un signe de pourcentage et un caractère (par exemple, %d). Si un signe de pourcentage est suivi d'un caractère sans signification pour la zone de format, ce caractère est copié dans STDOUT. Par exemple, pour imprimer un signe de pourcentage, tapez %%.

Les zones facultatives flags, width et precision figurant avant le caractère contrôlent d'autres aspects du format. Elles sont décrites ci-après :

flags
Ces *caractères facultatifs* contrôlent la justification de sortie et l'impression de signes, d'espaces, de séparateurs décimaux et de préfixes octo / hexadécimaux. Un spécificateur peut contenir plusieurs indicateurs.

Indicateurs possibles :

–
Signification : alignement à gauche du résultat dans la largeur de zone donnée.
Par défaut : alignement à droite.

+
Signification : fait précéder la valeur de sortie d'un signe (+ ou –) si elle est d'un type signé.
Par défaut : le signe apparaît uniquement pour les valeurs signées négatives (–).

0
Signification : si la largeur est précédée de 0, des zéros sont ajoutés jusqu'à atteindre la largeur minimum. Si 0 et – apparaissent, le zéro est ignoré. Si 0 est indiqué avec un format d'entier (i, u, x, X, o, d), il est ignoré.
Par défaut : pas de remplissage.

Espace (' ')
Signification : fait précéder la valeur de sortie d'un espace si elle est signée et positive ; l'espace est ignoré si les indicateurs espace et + apparaissent à la fois.
Par défaut : aucun espace.

#
Signification 1 : utilisé avec le type o, x, ou X, l'indicateur # fait précéder toute valeur de sortie non nulle de 0, 0x, ou 0X, respectivement.
Par défaut 1 : aucun préfixe n'apparaît.

Signification 2 : utilisé avec le type e, E, ou f, l'indicateur # force la valeur de sortie à contenir un séparateur décimal dans tous les cas.
Par défaut 2 : un séparateur décimal apparaît uniquement s'il est suivi de chiffres.

Signification 3 : utilisé avec le format g ou G, l'indicateur # force la valeur de sortie à contenir un séparateur décimal dans tous les cas et empêche la troncature de zéros en fin de valeur.

Par défaut 3 : un séparateur décimal apparaît uniquement s'il est suivi de chiffres. Les zéros en fin de valeur sont tronqués.
Remarque : ignorés quand ils sont utilisés avec d, i ou u.

width
Cette deuxième zone facultative, ou argument, contrôle le nombre minimum de caractères imprimés. Il s'agit d'un entier décimal non négatif.

- Si le nombre de caractères dans la valeur de sortie est inférieur à la largeur spécifiée, des espaces sont ajoutés à gauche ou à droite des valeurs — selon si – l'indicateur (pour l'alignement de gauche) est indiqué — jusqu'à atteindre la largeur minimum.
- Si la largeur est précédée de 0, des zéros sont ajoutés jusqu'à atteindre la largeur minimum (inutile pour les nombres alignés à gauche).
- La spécification de largeur n'entraîne jamais la troncature d'une valeur. Si le nombre de caractères dans la valeur de sortie est supérieur à la largeur spécifiée ou si celle-ci n'est pas mentionnée, tous les caractères sont imprimés (en fonction de l'indication de précision ci-après).

précision
Cette troisième zone facultative ou argument, désigne le nombre de caractères à imprimer, le nombre de positions décimales ou le nombre de chiffres importants. Contrairement à la spécification de largeur, celle de précision peut entraîner la troncature de la valeur de sortie ou arrondir une valeur à virgule flottante. Il s'agit d'un entier décimal non négatif précédé d'un point (.).

type
Ce caractère obligatoire détermine si l'argument associé est un entier, un double ou un point. La liste des types disponibles inclut :

d – entier décimal signé

i – entier décimal signé

o – entier octal non signé

u – entier décimal non signé

x – entier hexadécimal non signé, utilisant « abcdef »

X – entier hexadécimal non signé, utilisant « ABCDEF »

e – double de forme exponentielle [-]d.dddd e [signe]ddd

E – identique à e, sauf qu'il utilise E pour introduire un exposant

f – double sous la forme [-]dddd.dddd

g – formate au format e ou f, en fonction de celui qui est plus compact

G – identique à g, sauf que E est utilisé pour introduire l'exposant

Exemple de FORMAT :

Cet exemple illustre plusieurs instructions utilisant la fonction FORMAT dans une routine de mesure :

`ASSIGN/V1 = PROBEDATA("OFFSET")`	V1 devient de type point et représente les décalages du palpeur en cours. Avec les valeurs de la routine de mesure pour cet exemple, V1 devient : <-1,8898, 1,8898, 5,704>
`ASSIGN/V3 = FORMAT("%.5f, %.5f, %.5f", V1)`	V3 devient de type chaîne. La chaîne est formatée à l'aide de l'objet de point de la variable V1. V3 est alors : `-1.88976, 1.88976,`

	5.70403
`ASSIGN/V4 = 1.123456789`	V4 devient de type double.
`ASSIGN/V5 = FORMAT("%.5f ", V4) + FORMAT("%.6f ", V4) + FORMAT("%.7f ", V4) + FORMAT("%.8f", V4)`	V5 devient de type chaîne avec cette valeur : 1,12346 1,123457 1,1234568 1,12345679
`ASSIGN/V6A = "La valeur de V4 est : " + FORMAT("%.8f", V4)`	V6A devient un type chaîne avec la valeur : La valeur de V4 est : 1.12345679
`ASSIGN/V6B = FORMAT("La valeur de V4 est : %.8f", V4)`	Le résultat de l'expression reste identique à V6A ci-dessus.
`ASSIGN/V7 = 4444`	V7 devient de type double car tous les numéros sont supposés de ce type, sauf en cas de contrainte d'entier.
`ASSIGN/V8 = FORMAT("%o", INT(V7))`	V8 devient de type chaîne avec cette valeur : 10534
`ASSIGN/V9 = FORMAT("%u", INT(-1))`	V9 devient de type chaîne avec cette valeur : 4294967295
`ASSIGN/V10 = FORMAT("%x", INT(2143))`	V10 devient de type chaîne avec cette valeur : 85f
`ASSIGN/V11 = FORMAT("%X", INT(9567))`	V11 devient de type chaîne avec cette valeur : 255F
`ASSIGN/V12 = FORMAT("%e", 0.0005432)`	V12 devient de type chaîne avec cette valeur : 5,432000e-004
`ASSIGN/V13 = FORMAT("%E", 145.3421)`	V13 devient de type chaîne avec cette valeur : 1,453421E+002
`ASSIGN/V14 = FORMAT(",%6d,", INT(1))`	V14 devient de type chaîne avec cette valeur : `, 1,`
`ASSIGN/V15 = FORMAT(",%-6d,", INT(1))`	V15 devient de type chaîne avec cette valeur : `,1 ,`

GETSETTING

GETSETTING
Vous permet de renvoyer plusieurs réglages de PC-DMIS en fonction du paramètre de chaîne inséré.

GETSETTING(<*chaîne*>)

Vous pouvez utiliser ces paramètres de chaîne :

- "Mode CND" – Renvoie 1 si PC-DMIS est en mode CND, 0 dans les autres cas.
- "Mode manuel" – Renvoie 1 si PC-DMIS est en mode manuel, 0 dans les autres cas.
- "Alignement en cours" – Renvoie une chaîne de l'alignement en cours.
- "Plan de travail en cours" – Renvoie une chaîne du plan de travail en cours.
- "Valeur du plan de travail" – Renvoie une valeur numérique du plan de travail en cours.
- "Prépalpage" - Renvoie la valeur du prépalpage en cours comme numéro de précision double.
- "Recul" - Renvoie la valeur du recul en cours comme numéro de précision double.
- "Vérif" - Renvoie la valeur de la vérification en cours comme numéro de précision double.
- "Vitesse de contact" – Renvoie la valeur de la vitesse de contact en cours comme numéro de précision double.
- "Vitesse déplacement" - Renvoie la valeur de la vitesse de déplacement en cours comme numéro de précision double.
- "Mode Fly" – Renvoie 1 si PC-DMIS est en mode Fly, 0 dans les autres cas.
- "Ph9 présent" – Renvoie 1 si Ph9/Ph10 est présent, 0 dans les autres cas..
- "MMT manuelle" - Renvoie 1 si la MMT est manuelle, 0 dans les autres cas.
- "LangStr(<Nombre ou ID>)" – Renvoie une chaîne des ressources de PC-DMIS dans la langue en cours à partir d'un ID de ressource ou de l'un de ces ID : "Yes", "No", "Oper", "Rept", "Input", "Doc", "YesNo", "Readout", "Internal", "External", "Rect ", "Polr ", "Out", "In", "Least_Sqr", "Min_Sep", "Max_Insc", "Min_CircSc", "Fixed_Rad", "Workplane", "Xaxis", "YAxis", "ZAxis", "Xplus", "Xminus", "YPlus", "YMinus", "ZPlus", "ZMinus", "Point", "Plane", "Line", "Circle", "Sphere", "Cylinder", "Round_Slot", "Square_slot", "Cone", ou "None".

  Si la valeur utilisée est un nombre positif, PC-DMIS extrait la chaîne de son fichier resource.dll. Si vous utilisez un nombre négatif, PC-DMIS extrait la chaîne de son fichier strings.dll (tableau de chaînes).
- "Tôle étendue" – Renvoie 1 si la case **Afficher options de tôle étendues** est cochée dans la boîte de dialogue **Options de configuration**, 0 dans le cas contraire.
- "LastHitMove(X)" - Renvoie la valeur X de la commande HIT /BASIC ou MOVE/POINT la plus récente. PC-DMIS doit être en mode CND pour que ce paramètre fonctionne.
- "LastHitMove(Y)" - Renvoie la valeur Y de la commande HIT/BASIC ou MOVE/POINT la plus récente. PC-DMIS doit être en mode CND pour que ce paramètre fonctionne.

- "LastHitMove(Z)" - Renvoie la valeur Z de la commande HIT/BASIC ou MOVE/POINT la plus récente. PC-DMIS doit être en mode CND pour que ce paramètre fonctionne.

Pour déterminer si PC-DMIS est en mode CND ou manuel, observez l'exemple suivant, illustrant la fonction GETSETTING :

Exemples :

ASSIGN/DCCMODEVAR = GETSETTING("Mode CND"
) attribue à la variable DCCMODEVAR la valeur 1 si PC-DMIS est en mode CND, 0 dans les autres cas.

ASSIGN/DCCMODEVAR = GETSETTING("Mode manuel")
attribue à la variable MANMODEVAR la valeur 1 si PC-DMIS est en mode manuel, 0 dans les autres cas.

Pour déterminer le plan de travail en cours, prenez cet exemple :

Exemples :

ASSIGN/WORKPLANE_ID = GETSETTING("Plan de travail en cours")
attribue à la variable WORKPLANE_ID la valeur de chaîne du plan de travail en cours (ZPLUS, ZMINUS etc.).

ASSIGN/WORKPLANE_VALUE = GETSETTING("Valeur plan de travail")
attribue à la variable WORKPLANE_VALUE une valeur numérique représentant le plan de travail. Ces valeurs sont associées aux plans de travail : ZPLUS = 0, ZMOINS = 3, XPLUS = 1, XMOINS = 4, YPLUS = 2, ou YMOINS = 5.

GETTEXT

GETTEXT
Renvoie le texte en cours à partir du champ de données spécifié : `GETTEXT(<Chaîne ou Entier>, <Entier>, <Pointeur>)`

Cette fonction a trois champs.

Première zone - numéro de la zone de données ou description

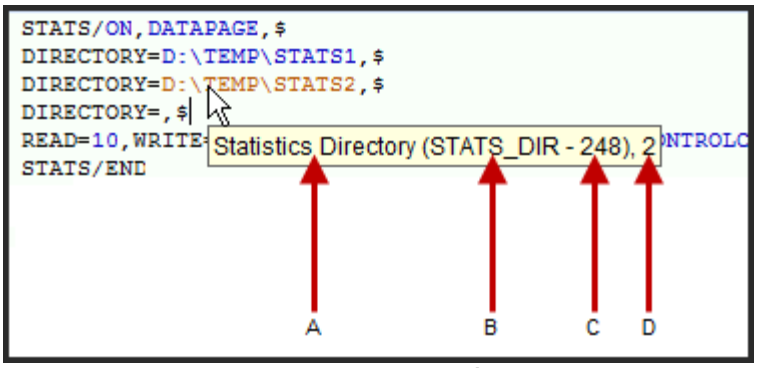
Le premier champ peut être une description de chaîne de la zone de données, indiqué par l'élément (A) dans l'image ci-dessous ou le numéro de la zone de données, indiqué dans l'élément (C) dans l'image ci-dessous.

Remarque : l'élément (B), dans l'image ci-dessous, n'est pas utilisé dans cette fonction, mais il est parfois utilisé en automatisation ou dans des expressions de rapports.

Pour obtenir ces valeurs :

1. Passez PC-DMIS en mode commande. Cliquez avec le bouton droit n'importe où dans la fenêtre de modification. Un menu de raccourcis s'affiche.
2. Dans ce menu de raccourci, sélectionnez **Modifier l'affichage de la fenêtre**, puis **Informations sur le type de données**.
3. Placez le pointeur de la souris sur une zone de données de la fenêtre de modification. La description de type, le numéro de type et l'index de type de cet élément de données s'affichent.

Remarque : la description de type pouvant varier selon la langue, si le programme est exécuté sur une version de PC-DMIS s'exécutant dans une autre langue, utilisez le numéro de type à la place.



Exemple d'information de type de données montrant (A) la description du type, (B) l'ID de la chaîne type, (C) le numéro de type et (D) l'index de type

Deuxième zone - Index de type

Le second champ est l'index de type, indiqué par (D) dans l'image ci-dessus. Ce champ est généralement fixé à zéro à moins que vous n'ayez plus d'instances du même type de champ dans la même commande telles que plusieurs champs RÉPERTOIRE illustrés dans l'image ci-dessus. Sa valeur correcte peut être obtenue en utilisant la même procédure que celle associée au premier.

Troisième zone - Pointeur de commande

Le troisième champ est un pointeur de commande. Il pointe vers la commande contenant le champ d'où le texte est extrait. Ce champ peut être spécifié à l'aide de la notation de pointage de commande (à savoir {F15}) ou de l'expression GETCOMMAND, comme indiqué dans cet exemple.

Exemples :

```
ASSIGN/V1 = GETTEXT("Type Math Best Fit", 0, {F15})
```

Cette commande affecte la valeur courante V1 de la bascule de type math best fit de l'élément F15.

```
ASSIGN/V2 = GETCOMMAND("Commentaire", "TOP", 1)

ASSIGN/V3 = GETTEXT("Type de commentaire", 1, V2)
```

V2 est affecté à un pointeur dans le premier commentaire à partir du début de la routine de mesure.

V3 reçoit la valeur du champ à bascule Type de commentaire. Si le premier commentaire de la routine de mesure doit être affiché à l'attention de l'opérateur, la valeur de V3 est "OPER".

Voir "Fonctions du pointeur" pour en savoir plus sur l'expression GETCOMMAND utilisée pour définir un pointeur pour une commande.

GETPROGRAMINFO

GETPROGRAMINFO
Renvoie les informations sur les routines de mesure en fonction des paramètres transférés dans : `GETPROGRAMINFO(<Chaîne>, <Chaîne facultative>)`

Cette fonction a au plus deux chaînes comme paramètres. Le plus souvent, vous n'avez besoin que du premier paramètre. Les champs de chaîne ne sont pas sensibles à la casse.

Premier champ—Chaîne

Le premier champ est une entrée de chaîne indiquant les informations à renvoyer.

DATE - Renvoie la date actuelle

TIME - Renvoie l'heure actuelle

PARTNAME - Renvoie le nom de la pièce tel que défini dans l'en-tête de la routine de mesure.

PARTPATH - Renvoie le chemin d'accès complet au fichier de la routine de mesure.

REVISION- Renvoie le numéro de révision tel que défini dans l'en-tête

DRAWING - Comme REVISION, renvoie le numéro de révision tel que défini dans l'en-tête

SERIALNUM - Renvoie le numéro de série tel que défini dans l'en-tête

STATSCOUNT - Renvoie le nombre statistique actuel

SEQNUM - Comme STATSCOUNT, renvoie le nombre statistique actuel

SHRINK - Renvoie le facteur d'échelle global

NUMMEAS - Renvoie le nombre de dimensions exécutées

NUMOOT - Renvoie le nombre de dimensions hors tolérance exécutées

ELAPSEDTIME - Renvoie le temps écoulé depuis le début de l'exécution

FILENAME - Renvoie le nom de fichier de la routine de mesure (.prg).

REPORTNAME - Renvoie le nom de fichier de sortie actuel

TEMP - Renvoie la température pour la seconde chaîne d'entrée facultative. Voir "Second champ—Chaîne facultative" ci-dessous.

PRGVERSION - Renvoie une valeur de chaîne du numéro de version PC-DMIS du fichier de routine de mesure. Vous pouvez enregistrer un fichier de routine de mesure pour qu'il soit compatible avec une version spécifique. Pour plus d'informations, voir "Enregistrer sous" au chapitre "Utilisation des options de fichier de base".

PRGSCHEMA - Renvoie un entier du numéro de schéma PC-DMIS du fichier de routine de mesure. Il s'agit d'une valeur interne utilisée par PC-DMIS pour indiquer les commandes et les options sérialisées.

PCDMISVERSION - Renvoie une valeur de chaîne de la version installée du logiciel PC-DMIS.

Second champ—Chaîne facultative

Le second champ est une entrée de chaîne facultative. Il est uniquement nécessaire si TEMP est pris comme première entrée. Les chaînes possibles ci-dessous proviennent de la commande de compensation de la température. Pour plus d'informations, voir "Compensation de la température" au chapitre "Définition des préférences".

TEMPP - Renvoie la température pour le capteur de la pièce

TEMPX - Renvoie la température pour le capteur de l'axe X

TEMPY - Renvoie la température pour le capteur de l'axe Y

TEMPZ - Renvoie la température pour le capteur de l'axe Z

REF_TEMP - Rnvoie la température de référence

HIGH_THRESHOLD - Renvoie le seuil supérieur de la température

LOW_THRESHOLD - Renvoie le seuil inférieur de la température

Exemple

```
$$ NO, Cet exemple de code montre le nombre total de dimensions et le nombre de
dimensions hors tolérance.
      ASSIGN/V1 = GETPROGRAMINFO("NUMMEAS")
```

```
        ASSIGN/V2 = GETPROGRAMINFO("NUMOOT")
        COMMENT/REPT
        "Total Dimensions: " + V1
        "Total Out of Tolerance: " + V2
$$ NO, Cet exemple de code renvoie la température pour le capteur de l'axe Z.
        ASSIGN/V3 = GETPROGRAMINFO("TEMP", "TEMPZ")
        COMMENT/REPT
        "Température de l'axe Z: " + V3
```

INDEX

INDEX
Emplacement de sous-chaîne *: INDEX(<chaîne>, <chaîne>)*

Renvoie l'emplacement de la seconde chaîne contenue dans la première. La première lettre de la chaîne est 1. Si la sous-chaîne est introuvable, la valeur renvoyée est zéro.

Pour voir un exemple simple de cette fonction, voir la rubrique "Exemple de code pour ligne en lecture" au chapitre "Utilisation d'entrées/sorties de fichier".

LEFT

LEFT
n caractères de chaîne restants : *LEFT(<Chaîne>, <Chaîne>)*

Renvoie une chaîne composée des caractères le plus à gauche spécifiés par la deuxième expression à partir de la chaîne spécifiée dans la première expression. La première expression devient automatiquement de type chaîne et la deuxième de type entier.

Pour voir un exemple simple de cette fonction, voir la rubrique "Exemple de code pour ligne en lecture" au chapitre "Utilisation d'entrées/sorties de fichier".

LEN

LEN
Longueur de chaîne : *LEN(<Chaîne>)*

Renvoie le nombre de caractères contenus dans la chaîne.

LOWERCASE

LOWERCASE
Crée une chaîne en minuscules : *LOWERCASE(<Chaîne>)*

Renvoie l'équivalent en minuscules de la chaîne.

MID

MID
n caractères restants au milieu : *MID(<Chaîne>, <Entier>, <Entier facultatif>)*

Renvoie une sous-chaîne composée des caractères de la chaîne spécifiée dans le premier paramètre à partir de la position spécifiée dans le deuxième pour n caractères spécifiés dans le troisième paramètre. Si le troisième paramètre n'est pas défini, le reste de la chaîne est renvoyé.

Pour voir un exemple simple de cette fonction, voir la rubrique "Exemple de code pour ligne en lecture" au chapitre "Utilisation d'entrées/sorties de fichier".

ORD

ORD
Conversion ordinale : *ORD(<Chaîne>)*

Renvoie l'entier de la valeur ASCII de la première lettre de la chaîne. (0-255).

PCDMISAPPLICATIONPATH

PCDMISAPPLICATIONPATH
Affichage du chemin complet : PCDMISAPPLICATIONPPATH*()*

Renvoie la chaîne contenant le chemin complet au répertoire de l'application où PC-DMIS est installé. Ce répertoire contient le principal exécutable et d'autres fichiers de programme nécessaires pour exécuter PC-DMIS.

PCDMISUSERHIDDENDATAPATH

PCDMISUSERHIDDENDATAPATH
Affichage du chemin complet : PCDMISUSERHIDDENDATAPATH*()*

Renvoie la valeur de chaîne contenant le chemin complet du répertoire de données utilisateur masqué et employé par PC-DMIS. Voir "Présentation des emplacements de fichiers" pour les fichiers figurant dans ce répertoire.

PCDMISUSERVISIBLEDATAPATH

PCDMISUSERVISIBLEDATAPATH
Affichage du chemin complet : PCDMISUSERHIDDENDATAPATH*()*

Renvoie la valeur de chaîne contenant le chemin complet du répertoire de données utilisateur visible et employé par PC-DMIS. Voir "Présentation des emplacements de fichiers" pour les fichiers figurant dans ce répertoire.

PCDMISSYSTEMHIDDENDATAPATH

PCDMISSYSTEMHIDDENDATAPATH
Affichage du chemin complet : PCDMISSYSTEMHIDDENDATAPATH*()*

Renvoie la valeur de chaîne contenant le chemin complet du répertoire de données système masqué et employé par PC-DMIS. Voir "Présentation des emplacements de fichiers" pour les fichiers figurant dans ce répertoire.

PCDMISSYSTEMVISIBLEDATAPATH

PCDMISSYSTEMVISIBLEDATAPATH
Affichage du chemin complet : PCDMISSYSTEMVISIBLEDATAPATH*()*

Renvoie la valeur de chaîne contenant le chemin complet du répertoire de données système visible et employé par PC-DMIS. Voir "Présentation des emplacements de fichiers" pour les fichiers figurant dans ce répertoire.

PCDMISSYSTEMREPORTINGPATH

PCDMISSYSTEMREPORTINGPATH
Affichage du chemin complet : PCDMISSYSTEMREPORTINGPATH*()*

Renvoie la valeur de chaîne contenant le chemin complet du répertoire de rapports employé par PC-DMIS. Ce répertoire contient les modèles de rapports et d'étiquettes utilisés par la fenêtre Rapport.

RIGHT

RIGHT
n caractères à droite de la chaîne : *RIGHT(<Chaîne>, <Entier>)*

Renvoie une chaîne composée des caractères le plus à droite spécifiés par l'entier de la chaîne.

SYSTEMDATE

SYSTEMDATE
Date système : SYSTEMDATE(<Chaîne de format de date>)

Renvoie la chaîne contenant la date courante. Par exemple, la commande SYSTEMDATE("MM'/'dd'/'aa") renvoie la chaîne "02/12/14" si la date actuelle le 12 février 2014.

Les composants de chaîne suivants servent à créer la chaîne de date. Veillez à respecter la casse indiquée (MM plutôt que mm). Les caractères autres que la date (comme les espaces) insérés entre les composants de la chaîne gardent dans la chaîne de sortie la place qu'ils occupaient dans celle d'entrée. Les caractères de la chaîne d'entrée qui sont délimités par des guillemets simples se retrouvent à la même place dans la chaîne de sortie, mais sans les guillemets.

d - Jour du mois en chiffre. Pas de zéro en tête pour les dates à un seul chiffre..

dd – Jour du mois (en chiffres). Zéro en tête pour les dates à un seul chiffre..

ddd – Jour de la semaine abrégé en trois lettres.

dddd – Jour de la semaine non abrégé.

M – Mois en chiffre, sans zéro en tête pour les mois à un seul chiffre.

MM – Mois en chiffre, avec zéro en tête pour les mois à un seul chiffre.

MMM – Mois abrégé en trois lettres.

MMMM – Mois non abrégé.

a – Année en chiffre, sans zéro en tête pour les années à un seul chiffre.

aa – Année en chiffre, avec zéro en tête pour les années à un seul chiffre.

aaaa – Année en quatre chiffres.

SYSTEMTIME

SYSTEMTIME
Heure système formatée : *SYSTEMTIME(<Chaîne du format d'heure>)*

Renvoie la chaîne contenant l'heure actuelle. Par exemple, la commande SYSTEMTIME("hh:mm:ss tt") renvoie l'heure dans une chaîne formatée comme "11:29:40 PM".

Les composants de chaîne suivants servent à créer la chaîne d'heure. Veillez à respecter la casse indiquée (**tt** au lieu de **TT**). Les caractères autres que l'heure (comme les espaces) insérés entre les composants de la chaîne du format d'heure gardent dans la chaîne de sortie la place qu'ils occupaient dans celle d'entrée. Les caractères de la chaîne d'entrée qui sont délimités par des guillemets simples se retrouvent à la même place dans la chaîne de sortie, mais sans les guillemets.

h - Heures sans zéro en tête pour les heures à un chiffre ; format 12 heures

hh – Heures avec un zéro en tête pour les heures à un chiffre ; format 12 heures

H - Heures sans zéro en tête pour les heures à un chiffre ; format 24 heures

HH – Heures avec un zéro en tête pour les heures à un chiffre ; format 24 heures

m – Minutes sans zéro en tête pour les minutes à un chiffre

mm – Minute avec un zéro en tête pour les minutes à un chiffre

s – Secondes sans zéro en tête pour les secondes à un chiffre

ss – Secondes avec un zéro en tête pour les secondes à un chiffre

t – Un caractère de marquage horaire (A ou P)

tt – Plusieurs caractères de marquage horaire (AM ou PM)

SYSTIME

SYSTIME
Heure système : *SYSTIME()*

Renvoie une chaîne composée de l'heure système actuelle. Cette fonction diffère de la fonction SYSTEMTIME décrite ci-dessus. Elle renvoie automatiquement le jour, la date, l'heure, suivie de l'année.

Exemple : "Wed February 12 13:50:21 2014"

Remarque : la chaîne de retour indiquant l'heure système est adaptée à l'heure locale.

UPPERCASE

UPPERCASE
Crée une chaîne en majuscules : UPPE*RCASE(<Chaîne>)*

Renvoie l'équivalent en majuscules de la chaîne.

Fonctions mathématiques

ABS

ABS Valeur absolue : *ABS(<Double>)*
Renvoie une valeur absolue d'entrée.

EXP

EXP Exponentielle : *EXP(<Double>)*
Renvoie la valeur exponentielle de l'expression.

LOG

LOG Base journal 10: *LOG(<Double>)*
Renvoie la base journal 10 de l'expression.

LN

LN Logarithme naturel : *LN(<Double>)*
Renvoie le logarithme naturel de l'expression.

ROUND

ROUND Arrondir : *ROUND(<Double>)*
Renvoie l'entrée arrondie à l'entier le plus proche.

SQRT

SQRT Racine carrée : *SQRT(<Double>)*
Renvoie la racine carrée de l'entrée.

Fonctions trigonométriques

Important : chaque fonction trigonométrique accepte et renvoie par défaut des radians. Pour des valeurs en degrés, utilisez la fonction RAD2DEG décrite ci-après.

ACOS

ACOS Cosinus arc : *ACOS(<Double>)*
Renvoie le cosinus de l'arc de l'expression. Par exemple, ACOS(5.0) renvoie 0. En général, ACOS(<expression>) renvoie le cosinus de l'arc de l'expression.

ASIN

ASIN Sinus arc : *ASIN(<Double>)*
Renvoie le sinus de l'arc de l'expression.

ATAN

ATAN Tangente arc : *ATAN(<Double>)*
Renvoie la tangente de l'arc de l'entrée.

COS

COS Cosinus : *COS(<Double>)*
Renvoie le cosinus de l'entrée.

DEG2RAD

DEG2RAD Degrés en radians : *DEG2RAD(<Double>)*
Renvoie l'entrée divisée par 360 et multipliée par 2π. Convertit les degrés en radians.

RAD2DEG

RAD2DEG Radians en Degrés: *RAD2DEG(<Double>)*
Renvoie l'entrée multipliée par 360 et divisée par 2π. Convertit les radians en degrés.

SIN

SIN Sinus : *SIN(<Double>)*
Renvoie le sinus de l'entrée.

TAN

TAN Tangente : *TAN(<Double>)*
Renvoie la tangente de l'entrée.

Remarque : les fonctions où l'entrée est hors limite (ce qui provoquerait une panne de l'ordinateur pour ACOS, ASIN, LOG, LN, SQRT, etc.) renvoient 0.

Fonctions de point

ANGLEBETWEEN

ANGLEBETWEEN

Angle entre : *ANGLEBETWEEN(<vecteur>, <vecteurr>)*

Renvoie l'angle en degrés formé par les deux vecteurs. Les deux paramètres doivent être des expressions évaluées en type de vecteur. Pour obtenir le vecteur d'un élément, vous devez utiliser l'ID d'élément suivi de l'extension .IJK. Voir l'exemple ci-dessous :

Exemple :

```
F1 =GENERIC/POINT,DEPENDENT,CARTESIAN,$
NOM/XYZ,<3,3,3>,$
MEAS/XYZ,<3,3,3>,$
NOM/IJK,<1,0,0>,$
MEAS/IJK,<1,0,0>
F2 =GENERIC/POINT,DEPENDENT,CARTESIAN,$
NOM/XYZ,<10,10,10>,$
MEAS/XYZ,<10,10,10>,$
NOM/IJK,<0,0,1>,$
MEAS/IJK,<0,0,1>
ASSIGN/V1=F1.IJK
ASSIGN/V2=F2.IJK
ASSIGN/V3=ANGLEBETWEEN(V1,V2)
COMMENT/OPER,NO,FULL SCREEN=NO,AUTO-CONTINUE=NO,
"L'angle entre " + V1 + " et " + V2 + " est : " + V3
```

CROSS

CROSS
Produit croisé : *CROSS(<Point>, <Point>)*

La valeur de renvoi est de type point et constitue le produit vectoriel des deux expressions.

DELTA

DELTA
Décalage de vecteur *: DELTA(<Point>, <Point>, <Double>)*

La fonction considère la première expression (point) et calcule un nouveau point dans la direction de la seconde expression (vecteur) en décalage par rapport à la troisième expression. Par exemple, `DELTA(MPOINT(0,0,0), MPOINT(1,0,0), 10)` renvoie le point 10,0,0.

DOT

DOT
Produit de points : *DOT(<Point>, <Point>)*

Renvoie le produit scalaire des deux points (vecteurs).

UNIT

UNIT
Vecteur d'unité : *UNIT(<Point>)*

Renvoie le point divisé par sa longueur. Par exemple, `UNIT(MPOINT(0,0,0))` renvoie le point 0,0,1.

MPOINT

MPOINT
Contrainte de point : *MPOINT(<Expression1>, <Expression2>, <Expression3>)*

Contraint les trois expressions à un type Point.

Exemple :
```
ASSIGN/V1 = MPOINT(2.5,3.6,4)
```

V1.X a une valeur de 2,5
V1.Y a une valeur de 3,6
V1.Z a une valeur 4,0

Voir "Contrainte de point".

Fonctions de pointeur

DIST2D

DIST2D
Distance 2D : DIST2D(<FEAT1>, <FEAT2>, <FEAT3>)

Les éléments doivent se trouver entre accolades.

Calcule la distance entre les deux premiers arguments dans la commande (élém1 et élém2), perpendiculaires au troisième (élém3).

- Si le troisième argument est un plan, la distance entre les deux premiers est calculée perpendiculaire à ce plan.
- Si le troisième argument est une droite ou un cylindre, la distance entre les deux premiers est calculée perpendiculaire au troisième argument dans le plan de travail actif.

Par exemple, si le plan XY est le troisième argument, il a un vecteur Z+ (0,0,1) et la distance indiquée est uniquement dans l'axe Z.

Exemple

```
ASSIGN/V3=DIST2D({CIR1},{CIR2},{PLN1})
COMMENT/OPER,NO,FULL SCREEN=NO,AUTO-CONTINUE=NO,
V3
```

DIST3D

DIST3D
Distance 3D : DIST3D(<FEAT1>, <FEAT2>)

Calcule la distance 3D entre les deux éléments.

Remarque : les éléments doivent se trouver entre accolades.

Exemple

```
ASSIGN/V3=DIST3D({CIR1},{CIR2})
COMMENT/OPER,NO,FULL SCREEN=NO,AUTO-CONTINUE=NO,
V3
```

GETCOMMAND

GETCOMMAND
obtient un pointeur à la commande indiquée par les paramètres : `GETCOMMAND(<entier ou chaîne>, <chaîne>, <entier>`

Premier paramètre – Champ d'information sur la commande

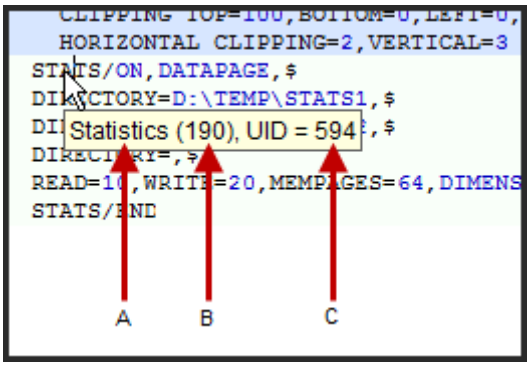
Le premier paramètre correspond au champ d'informations sur la commande. Il définit le type de commande à rechercher. L'élément suivant peut être transmis :

- Une chaîne de description de commande. Voir (A) dans le graphique ci-dessous.
- Un numéro de type de commande. Voir (B) dans le graphique ci-dessous.
- L'identificateur numérique unique. Voir (C) dans le graphique ci-dessous.

Si l'ID unique de la commande est transmis, aucun autre argument n'est nécessaire.

Pour obtenir la chaîne de description de commande, le numéro de type de commande et l'ID numérique unique :

1. Cliquez avec le bouton droit dans la fenêtre Édition.
2. Choisissez **Modifier l'affichage de la fenêtre | Information sur la commande** (PC-DMIS doit être en mode commande).
3. Placez le pointeur de la souris sur la commande appropriée. La description de la commande, le numéro de type et l'ID numérique unique s'affichent.



Exemple d'information de commande affichant (A) la chaîne de description de commande, (B) le numéro de type de commande et (C) l'ID numérique unique (UID)

Deuxième paramètre – Sens de la recherche

Le second paramètre correspond à la direction de la recherche. Les valeurs autorisées sont :

Valeur	Description
HAUT	Cette valeur indique que la recherche doit commencer à partir de la commande courante et se poursuivre vers le haut.
BAS	Cette valeur indique que la recherche doit commencer à partir de la commande courante et se poursuivre vers le bas.
DÉBUT	Cette valeur indique que la recherche doit commencer au début de la routine de mesure et se poursuivre vers le bas.
FIN	Cette valeur indique que la recherche doit commencer à partir du dernier objet de la routine de mesure et se poursuivre vers le haut.

Troisième paramètre – Instance à rechercher

Le troisième paramètre indique quelle instance de la commande doit être trouvée si plusieurs instances de la même commande sont dans la routine de mesure.

Exemple : si la routine de mesure a deux instances d'une commande STATS/ACTIVER et que vous voulez obtenir un pointeur vers la seconde instance à partir du haut, "2" est transmis comme troisième paramètre et "TOP" comme deuxième paramètre, comme indiqué ici.

```
ASSIGN/V1 = GETCOMMAND("Statistiques", "TOP", 2)
```

La fonction GETCOMMAND peut être utilisée pour fournir le troisième paramètre à la fonction de chaîne GETTEXT. Voir "Fonctions de chaîne" pour en savoir plus sur GETTEXT.

LEN

LEN
Nombre de boucles de pointeur : *LEN(<POINTER>)*

Renvoie le nombre de boucles d'un pointeur. Par exemple, si l'élément CIR1 est dans une boucle qui se répète 10 fois, vous pouvez mémoriser le nombre de fois que CIR1 a été mesuré dans une variable, en utilisant une instruction ASSIGN telle que :
`ASSIGN/V1=LEN({CIR1})`

Fonctions de tableau

ARRAY

ARRAY: Création du tableau : ARRAY(<EXPRESSION1>, <EXPRESSION2>, <EXPRESSION3>, …)
Crée un objet avec les éléments du tableau indiqués par des paramètres d'expression. Les éléments du tableau sont numérotés avec un index de base 1.

EQUAL

EQUAL : Comparaison de tableaux élément par élément : *EQUAL(<ARRAY>, <ARRAY>)*
Compare les deux tableaux élément par élément s'ils ont le même contenu. Si les deux tableaux ne sont pas de la même taille ou si les éléments de l'un n'ont pas d'éléments correspondants dans l'autre, la fonction renvoie un zéro. Dans les autres cas, la valeur de retour est 1.

LEN

LEN : Nombre d'éléments d'un tableau : *LEN(<ARRAY>)*
Renvoie le nombre d'éléments dans le tableau.

MAX

MAX : Plus grand élément du tableau : *MAX(<ARRAY>)*
Renvoie le plus grand élément du tableau. Les différents éléments du tableau font l'objet d'une comparaison numérique ou alphabétique.

MIN

MIN : Plus petit élément du tableau : *MIN(<ARRAY>)*
Renvoie le plus petit élément du tableau. Les différents éléments du tableau font l'objet d'une comparaison numérique ou alphabétique.

Fonctions diverses

ARCSEGMENTENDINDEX

ARCSEGMENTENDINDEX
renvoie le numéro d'index du point final d'un segment d'arc indiqué à partir d'un scanning : ARC*SEGMENTENDINDEX(<ID>, <index>,<tol1>,<tol2>)*

<ID> – Le premier paramètre est une valeur de chaîne de l'ID du scanning sur lequel la fonction extrait le numéro d'index du point final pour l'arc. Il peut s'agir de l'ID entre guillemets ou de toute expression qui, contrainte à la chaîne type, correspond finalement à l'ID d'un scan.

<index> – Le deuxième paramètre est le numéro d'index pour l'arc depuis lequel vous voulez obtenir le numéro du point final. Il s'agit d'une valeur de base 1. Par exemple, le numéro d'index de l'arc serait 3 si vous voulez le numéro du point final pour le troisième arc dans le scan.

<tol1> – Le troisième paramètre est la tolérance d'élément générale. Il s'agit d'une erreur de forme maximum servant à découper le scanning en droites et en arcs.

<tol2> – Le quatrième paramètre est la tolérance affinée. En général, cette tolérance plus réduite sert à placer des points depuis une extrémité de l'élément, jusqu'à ce que l'erreur de forme du segment se trouve dans cette tolérance.

Lorsque vous avez les index de début et de fin pour un arc, vous pouvez utiliser ces points dans un élément construit pour construire un arc distinct. Voir « Exemple de droite créée à partir d'un segment de scan » pour un exemple similaire.

ARCSEGMENTSTARTINDEX

ARCSEGMENTSTARTINDEX
renvoie le numéro d'index du point de départ d'un segment d'arc indiqué à partir d'un scanning : ARC*SEGMENTSTARTINDEX(<ID>, <index>,<tol1>,<tol2>).*

<ID> – Le premier paramètre est une valeur de chaîne de l'ID du scan sur lequel la fonction extrait le numéro d'index du point de départ pour l'arc. Il peut s'agir de l'ID entre guillemets ou de toute expression qui, contrainte à la chaîne type, correspond finalement à l'ID d'un scan.

<index> - Le deuxième paramètre est le numéro d'index pour l'arc depuis lequel vous voulez obtenir le numéro du point de départ. Il s'agit d'une valeur de base 1. Par exemple, le numéro d'index de l'arc serait 3 si vous voulez le numéro du point de départ pour le troisième arc dans le scan.

<tol1> – Le troisième paramètre est la tolérance d'élément générale. Il s'agit d'une erreur de forme maximum servant à découper le scanning en droites et en arcs.

<tol2> – Le quatrième paramètre est la tolérance affinée. En général, cette tolérance plus réduite sert à placer des points depuis une extrémité de l'élément, jusqu'à ce que l'erreur de forme du segment se trouve dans cette tolérance.

Il existe deux autres paramètres qui déterminent si un segment d'arc identifié dans un scan est acceptable. Ils sont uniquement modifiables avec l'Éditeur de réglages PC-DMIS. Tout segment d'arc ayant un rayon inférieur à `MinimumArcSegmentRadiusInMM` est rejeté. La valeur par défaut pour ce paramètre est de 2 mm. De la même façon, tout segment d'arc ayant un rayon supérieur à `MaximumArcSegmentRadiusInMM` est rejeté. La valeur par défaut pour ce paramètre est de 2000 mm (il ne doit pas être nécessaire de changer cette valeur).

Lorsque vous avez les index de début et de fin pour un arc, vous pouvez utiliser ces points dans un élément construit pour construire un arc distinct. Voir « Exemple de droite créée à partir d'un segment de scan » pour un exemple similaire.

EOF et EOL

EOF et **EOL** :
pour en savoir plus sur ces fonctions, voir la rubrique "Vérification de la fin d'un fichier ou d'une droite" au chapitre "Utilisation d'entrées/sorties de fichier".

FUNCTION

FUNCTION
Crée une fonction **:** FUNCTION((<PARAM1>, <PARAM2>…), <EXPRESSION>)

Crée une fonction prenant le nombre de paramètres indiqué par la liste de paramètres et substitue ces paramètres dans l'expression.

- Le premier élément à considérer lors de l'utilisation du mot-clé `FUNCTION` est la liste des paramètres.
- Cette liste se compose de noms de paramètres séparés par des virgules.
- La liste des paramètres est également entre parenthèses.
- Le deuxième élément correspond à l'expression.
- L'expression comprend les noms des paramètres qui sont remplacés par les paramètres eux-mêmes lorsque la fonction est appelée.

Voir « Exemple de fonction générique » à titre d'exemple.

GETROTABDATA

GETROTABDATA
Retourne les données à partir de la table tournante : GETROTABDATA*(<Paramètre>)*

<Paramètre> - Il peut s'agir de "Centre" ou "Angle".

"Centre" - Renvoie le centre XYZ de la table.

"Angle" - Renvoie l'angle actuel de la table.

Exemples :

`ASSIGN/V1 = GETROTABDATA("CENTER")`	V1 est défini avec la valeur du centre XYZ de la table tournante.
`ASSIGN/V2 = GETROTABDATA("ANGLE")`	V2 est défini avec la valeur de l'angle actuel XYZ de la table tournante.

IF

IF
Évaluation de l'expression conditionnelle : *IF(<EXPRESSION1>, <EXPRESSION2>, <EXPRESSION3>)*

Si l'expression1 est vraie (non nulle), cette fonction renvoie la valeur de l'expression 2 ; dans les autres cas, elle renvoie la valeur de l'expression 3.

ISIOCHANNELSET

ISIOCHANNELSET
Cette expression accepte deux paramètres. Le premier indique le canal d'E/S qui sera vérifié (la plage de numéros disponible dépend de la machine employée). Le détermine si le logiciel demandera le bras 1 ou le bras 2 de la machine. Si le second paramètre est défini à 1 (un), il demandera le contrôleur du bras 2. Si le second paramètre n'est pas présent (ou s'il est défini à zéro), le canal d'E/S demander le contrôleur du bras 2. Le contrôleur du bras 1 est la seule option si vous n'est pas en mode maître/esclave.

Remarque : si le type de données du palpeur, l'ID du contact, le nom de fichier du palpeur ou le numéro de canal sont incorrects, l'expression donne 0 comme résultat.

Exemple :

`ASSIGN/V4 = ISIOCHANNELSET(3, 0)`	V4 est égal à 1 (vrai) lorsque le canal est défini, sinon il est égal à 0 (faux).

LINESEGMENTENDINDEX

LINESEGMENTENDINDEX
renvoie le numéro d'index du point final d'un segment de droite indiqué à partir d'un scanning : LINE*SEGMENTENDINDEX(<ID>, <index>,<tol1>,<tol2>).*

<ID> – Le premier paramètre est une valeur de chaîne de l'ID du scanning sur lequel la fonction extrait le numéro d'index du point final pour le segment de droite. Il peut s'agir de l'ID entre guillemets ou de toute expression qui, contrainte à la chaîne type, correspond finalement à l'ID d'un scan.

<index> - Le deuxième paramètre est le numéro d'index pour le segment de droite depuis lequel vous voulez obtenir le numéro du point final. Il s'agit d'une valeur de base 1. Par exemple, le numéro d'index du segment de droite serait 3 si vous voulez le numéro du point final pour la troisième droite dans le scan.

<tol1> – Le troisième paramètre est la tolérance d'élément générale. Il s'agit d'une erreur de forme maximum servant à découper le scanning en droites et en arcs.

<tol2> – Le quatrième paramètre est la tolérance affinée. En général, cette tolérance plus réduite sert à placer des points depuis une extrémité de l'élément, jusqu'à ce que l'erreur de forme du segment se trouve dans cette tolérance.

Lorsque vous avez les index de début et de fin pour un segment de droite, vous pouvez utiliser ces points dans un élément construit pour construire une droite distincte. Voir « Exemple de droite créée à partir d'un segment de scan » pour un exemple similaire.

LINESEGMENTSTARTINDEX

LINESEGMENTSTARTINDEX
renvoie le numéro d'index du point de départ d'un segment de droite indiqué à partir d'un scanning : LINE*SEGMENTSTARTINDEX(<ID>, <index>,<tol1>,<tol2>).*

<ID> – Le premier paramètre est une valeur de chaîne de l'ID du scan sur lequel la fonction extrait le numéro d'index du point de départ pour le segment de droite. Il peut s'agir de l'ID entre guillemets ou de toute expression qui, contrainte à la chaîne type, correspond finalement à l'ID d'un scan.

<index> - Le deuxième paramètre est le numéro d'index pour le segment de droite depuis lequel vous voulez obtenir le numéro du point de départ. Il s'agit d'une valeur de base 1. Par exemple, le numéro d'index du segment de droite serait 3 si vous voulez que le numéro du point de départ pour la troisième droite soit dans le scanning.

<tol1> – Le troisième paramètre est la tolérance d'élément générale. Il s'agit d'une erreur de forme maximum servant à découper le scanning en droites et en arcs.

<tol2> – Le quatrième paramètre est la tolérance affinée. En général, cette tolérance plus réduite sert à placer des points depuis une extrémité de l'élément, jusqu'à ce que l'erreur de forme du segment se trouve dans cette tolérance.

Il existe un autre paramètre qui détermine si un segment de droite identifié dans un scan est acceptable. Il est uniquement modifiable avec l'éditeur de réglages PC-DMIS. Tout segment de droite d'une longueur inférieure à `MinimumLineSegmentLengthInMM` est rejeté. La valeur par défaut pour ce paramètre est de 2 mm.

Lorsque vous avez les index de début et de fin pour un segment de droite, vous pouvez utiliser ces points dans un élément construit pour construire une droite distincte. Voir « Exemple de droite créée à partir d'un segment de scan » pour un exemple similaire.

PROBEDATA

PROBEDATA
renvoie des données sur le palpeur actuel ou indiqué :
PROBEDATA(<OPTPROBEDATATYPE>, <OPTTIPID>, <OPTPROBEFILENAME>)

Cette fonction accepte jusqu'à trois paramètres facultatifs. Vous devez uniquement insérer des virgules entre les paramètres si vous en utilisez plusieurs. Il est inutile d'utiliser des virgules entre des paramètres vides. Par exemple, pour obtenir le diamètre du palpeur actuel, il suffit d'utiliser `ASSIGN/V1 = PROBEDATA('DIAM")`.

OPTPROBEDATATYPE - Paramètre facultatif spécifiant les données du palpeur que l'expression doit renvoyer. Si ce paramètre n'est pas fourni, l'ID du contact courant est renvoyée. Ce paramètre est de type chaîne. Toute expression donnant comme résultat une expression de chaîne valide peut être insérée dans la première expression. Les expressions de chaînes valides (sans distinction de majuscule/minuscule) pour le premier paramètre, incluent les expressions suivantes. Il s'agit d'expressions de chaînes qui doivent être entre guillemets doubles :

*"**Décalage**"* – Décalage X,Y,Z mesuré du contact. Renvoie un type point.

*"**Vecteur**"* – Vecteur de contact. Renvoie un type point.

*"**A**"* - Angle de contact A. Renvoie un type double.

*"**B**"* – Angle de contact B. Renvoie un type double.

*"**Diam**" ou **"Diamètre**"* - Diamètre du contact mesuré. Les quatre premières lettres "Diam" sont obligatoires, mais vous pouvez en inclure davantage, voire le nom complet "Diamètre". Renvoie un type double.

*"**Épais**" ou **"Épaisseur**"* - Épaisseur du contact mesurée. Les cinq premières lettres "Épais" sont obligatoires, mais vous pouvez en inclure davantage, voire le nom complet "Épaisseur". Renvoie un type double.

*"**Date**"* – Date de la dernière qualification du contact. Renvoie un type chaîne.

"Heure" – Heure de la dernière qualification du contact. Renvoie un type chaîne.

*"**ID**"* – ID de contact. Paramètre par défaut. Renvoie un type chaîne.

"PrbRdv" – Déviation radiale du palpeur. Renvoie un type double.

*"**Standarddeviation**"* – Déviation standard des palpeurs. Renvoie un type double.

*"**C**"* - Angle C d'un positionneur de palpeur CW43 léger. Renvoie un entier type.

Remarque : l'ajout d'un "**T**" devant "**Décalage**", "**Diamètre**" ou "**Épaisseur**" renvoie les informations théoriques (par exemple, TOFFSET, TDIAMETER et TTHICKNESS).

OPTTIPID - Ce paramètre facultatif indique le contact à utiliser pour obtenir les données de palpeur spécifiées dans la première expression. Si ce paramètre n'est pas fourni, le contact courant est utilisé. Ce paramètre doit être de type chaîne.

OPTPROBEFILENAME - Ce paramètre facultatif spécifie le nom de fichier du palpeur à utiliser pour obtenir les données du palpeur. Si ce paramètre n'est pas fourni, le fichier du palpeur courant est utilisé.

Exemples :

`ASSIGN/V1 = PROBEDATA()`	V1 est défini sur l'ID du contact courant ("T1A0B0").
`ASSIGN/V2 = PROBEDATA("TOFFSET", "T1A45B0")`	V2 est défini sur le décalage théorique du palpeur pour le contact T1A45B0.
`ASSIGN/V3 = PROBEDATA("Date", "T1A90B90", "MYPROB")`	V3 est défini sur une chaîne représentant la date de la dernière qualification de la pointe de contact T1A90B90 du fichier du palpeur MYPROB.

TUTORELEMENT

TUTORELEMENT: cette fonction accepte un argument, qu'il soit un nombre ou une chaîne (une chaîne serait l'ID d'un élément).

Cette fonction fonctionne avec le type de variable *Structures*. Voir « Structures » pour obtenir des explications relatives à la structure et aux sous-éléments.

Exemples :

`ASSIGN/E = TUTORELEMENT(1)`	Crée une structure Tutor Element.
`ASSIGN/WM = TUTORELEMENT(n)`	Pour tout nombre supérieur à 1, crée un tableau de n structures Tutor Element.
`ASSIGN/CIR1E = TUTORELEMENT("CIR1")`	Copie les données de l'élément CER1 dans les structures TutorElement.

La structure TutorElement possède actuellement les sous-éléments suivants :

Sous-élément	Description
ID	Chaîne de l'ID d'élément
TYPE	ENTIER (TYPEF)

X, Y, Z	Valeurs des coordonnées X, Y et Z
RP	Rayon polaire
AP	Angle polaire
CX	I
CY	J
CZ	K
DM	Diamètre 1
DM2	Diamètre 2
DS	Distance à partir de l'origine
A	Angle
AXY	Angle dans le plan XY
AYZ	Angle dans le plan YZ
AZX	Angle dans le plan ZX
F	Erreur de forme
SDEV	Écart type
TP	Position

QUALTOOLDATA

QUALTOOLDATA: cette fonction renvoie des données sur l'outil de calibrage en cours ou spécifié. Elle a la syntaxe suivante :

QUALTOOLDATA(<TOOLINFO>, <TOOLID>, <FACENUMBER>)

Cette fonction accepte jusqu'à trois paramètres. Elle a besoin d'au moins un paramètre pour renvoyer des données :

Le premier paramètre, *<TOOLINFO>,* est une *chaîne* indiquant le type d'informations à renvoyer sur l'outil de calibrage. Si vous ne transmettez pas ce paramètre, cette fonction renvoie le nom de l'outil de calibrage en cours ou spécifié.

- "***DIAM***" – Renvoie le diamètre de l'outil en tant que valeur de double.
- "***ID***" – Renvoie le nom de l'outil en tant que valeur de chaîne.
- "***LENGTH***" - Identique à "DIAM". Il renvoie également le diamètre de l'outil comme valeur double.
- "***OVERRIDEIJK***" - Renvoie le vecteur IJK de remplacement comme valeur de point.
- "***POLYDIAM***" - Renvoie le diamètre de la face polyhédrique indiquée comme valeur de double.
- "***POLYIJK***" - Renvoie le vecteur IJK de la face polyhédrique spécifiée comme valeur de point.
- "***POLYXYZ***" - Renvoie le centre XYZ de la face polyhédrique spécifiée comme valeur de point.
- "***SHANKIJK***" - Renvoie le vecteur IJK de la tige comme valeur de point.
- "***TYPE***" - Renvoie le type d'outil comme valeur d'entier (0 pour une sphère, 1 pour une sphère de Bras2, 2 pour un polyèdre, 3 pour un polyèdre de Bras2).
- "***WIDTH***" – Ce paramètre n'est plus utilisé.
- "***XYZ***" – Renvoie l'emplacement XYZ de l'outil comme valeur de point.

Le deuxième paramètre *<TOOLID>* est une *chaîne* désignant le nom de l'outil de calibrage pour lequel l'utilisateur souhaite recevoir des informations. Si vous ne transmettez pas ce paramètre, PC-DMIS suppose que vous souhaitez des informations sur l'outil de calibrage en cours. La chaîne n'est pas sensible à la casse.

Le troisième paramètre, *<FACENUMBER>*, est uniquement requis lorsque vous travaillez avec un outil de calibrage polyhédrique et lorsque le premier paramètre est « POLYXYZ », « POLYIJK » ou « POLYDIAM ». Il s'agit d'une valeur d'entier désignant la face de l'outil polyhédrique à utiliser pour obtenir des données.

Exemples :

`ASSIGN/VDIAM = QUALTOOLDATA("DIAM","SPHERE_1_IN")`	Attribue à la variable `VDIAM` le diamètre de l'outil SPHÈRE_1_IN.
`ASSIGN/VID = QUALTOOLDATA("ID")`	Attribue à la variable `VID` le nom de l'outil en cours.
`ASSIGN/VTYPE = QUALTOOLDATA("TYPE")`	Attribue à la variable `VTYPE` le type de l'outil en cours.
`ASSIGN/VPOLYDIAM = QUALTOOLDATA("POLYDIAM","POLYTEST",3)`	Attribue à la variable `VPOLYDIAM` le diamètre de la face 3 sur l'outil polyhédrique POLYTEST.

Exemples de fonctions

Ci-après différents exemples de fonctions pouvant permettre de créer et d'utiliser vos propres fonctions :

- Exemple de fonction générique
- Exemples de fonctions transmises comme variables
- Exemple de fonction avec plusieurs paramètres
- Exemple de fonctions créant d'autres fonctions
- Exemple de fonctions comme membres d'un tableau
- Exemple de fonctions définies de façon récursive

Exemple de fonction générique

`Assign/MYFUNC = FUNCTION((X,Y,Z), X*3 + Y*2 + Z)`
Crée une fonction définie par l'utilisateur et l'affecte à la variable MYFUNC. La fonction accepte trois paramètres : X, Y et Z.

X est multiplié par 3.

Y est multiplié par 2.

Z conserve simplement la valeur transmise.

Le total de X + Y + Z est ce qui est renvoyé.

`Assign/V1 = MYFUNC(7,2,5)`
Affecte à V1 la valeur 30 en évaluant les paramètres transférés dans la fonction MYFUNC(7,2,5). 7 est le paramètre substitué à X lorsque X apparaît dans la partie de l'expression appartenant à la définition de la fonction. Par conséquent, X*3 devient 7*3, soit 21.

2 est remplacé là où figure Y, donc Y*2 donne 2*2, ou 4.

5 est remplacé là où figure Z.

Les valeurs sont ensuite toutes ajoutées (21 + 4 + 5) et transmises à V1.

Exemples de fonctions transmises comme variables

Les fonctions peuvent être transmises en tant que variables. L'exemple suivant s'inspire de l'exemple de fonction générique ci-dessus :

`Assign/NEWFUNC = MYFUNC`
Définit la variable NEWFUNC pour qu'elle ait la même fonction que MYFUNC.

`Assign/V3 = NEWFUNC(12,2,3)`
Force V3 à avoir la valeur 43 à partir des expressions évaluées dans la fonction (36 + 4 + 3).

Exemple de fonction avec plusieurs paramètres

Les fonctions peuvent avoir plusieurs paramètres :

`Assign/ADDANDDOUBLE = FUNCTION((A,B), 2*(A+B))`
Crée une fonction et l'affecte à la fonction ADDANDDOUBLE. La fonction prend deux paramètres, les ajoute puis multiplie le résultat par 2.

`Assign/V2 = ADDANDDOUBLE(4, 5)`
Affecte à V2 la valeur 18. Les paramètres 4 et 5 sont remplacés dans la partie de l'expression de la fonction, ce qui donne 2*(4+5).

Exemples de fonctions créant d'autres fonctions

Des fonctions peuvent créer d'autres fonctions.

`Assign/COMPOSE = FUNCTION((F, G), FUNCTION((X), G(F(X)) ))`
Force COMPOSE à être une fonction en en prenant deux autres comme paramètres et crée une nouvelle fonction à partir des deux.

`Assign/ADD2 = FUNCTION((X), X+2)`
Force ADD2 à être une fonction qui ajoute deux au paramètre transféré.

```
Assign/ADD3 = FUNCTION((X), X+3)
```
Force ADD3 à être une fonction qui ajoute trois au paramètre transféré.

Assign/ADD5 = COMPOSE(ADD2, ADD3)
Force ADD5 à être une fonction composée des fonctions ADD2 et ADD3.

```
Assign/V5 = ADD5(3)
```
Force V5 à avoir la valeur V8.

Exemples de fonctions comme membres d'un tableau

Les fonctions peuvent être membres d'un tableau.

```
Assign/ANARRAY = ARRAY(3, FACTORIAL, "Hello World", ADD5)
```
Attribue ANARRAY à un tableau de 4 éléments : un nombre(3), une fonction (FACTORIAL), une chaîne ("Hello World") et une fonction (Add5).

```
Assign/V6 = ANARRAY[2](4)
```
Le second élément de ANARRAY est la fonction FACTORIAL. Le paramètre 4 est transféré à cette fonction et le résultat, soit 24, est affecté à V6.

```
Assign/V7 = ANARRAY[2](ANARRAY[4] (ANARRAY[1]))
```
De l'intérieur vers l'extérieur : le premier élément de ANARRAY (3) est transféré à la fonction du quatrième élément du tableau (Add5). Le résultat, soit 8, est transféré à la fonction du deuxième élément de tableau (FACTORIAL) et affecté à V7. V7 reçoit la valeur 40320.

Exemples de fonctions définies de façon récursive

Les fonctions peuvent être définies de façon récursive (pour s'appeler elles-mêmes).

```
Assign/FACTORIAL = FUNCTION((X), IF(X<=1, 1, X*FACTORIAL(X-1))
```
Crée une fonction dite factorielle et acceptant un paramètre. Si le paramètre est inférieur ou égal à 1, il est évalué à 1 ; dans le cas contraire, il est évalué à X multiplié par la fonction FACTORIAL de X-1.

```
Assign/V4 = FACTORIAL(5)
```
Affecte à V4 la valeur 120 (5*4*3*2*1).

Exemple de droite créée à partir d'un segment de scan

Cette rubrique fournit un exemple d'utilisation du langage d'expression PC-DMIS, en particulier les fonctions de segment de droite, en vue d'exporter des numéros de points de départ et de fin pour des segments de droite dans un scan, puis de créer votre propre élément de droite à l'aide des points extraits dans un élément construit. Vous pouvez appliquer les mêmes principes présentés dans cet exemple pour créer un segment d'arc à partir d'un scan.

Imaginez que votre routine de mesure comprend un élément de scan nommé SCN1 qui ressemble à ceci :

```
SCN1=FEAT/SCAN,LINEAROPEN,SHOW HITS=NO,SHOWALLPARAMS=YES
      MODE EXEC=RÉAPPRENDRE, MODE VALNOMS=RECHERCHE NOMS,PLANSÉCURITÉ=NON,POINT
      UNIQUE=NON,ÉPAISSEUR=0
      RECHERCHEVALNOMS=5,SEULEMENTSÉLECTIONNÉ=NON,UTILISERBESTFIT=NON,COMPPALPEUR=OUI,DÉP
      LACER ÉVITEMENT=NON,DISTANCE=0,Compensation CAO=NON
      DIR1=VARIABLE,
      TYPEPALPAGE=VECTEUR
      INITVEC=0,-1,0
      DIRVEC=1,0,0
      VECCOUPE=0,0,1
      FINVEC=0,-1,0
      VECPLAN=-1,0,0
      POINT1=100,0,-5
      POINT2=70,0,-5
      MESU/SCAN
      SCANDEBASE/DROITE,AFFICHER PALPAGES=NON,AFFICHERTOUSPARAMS=OUI
      <100,0,-5>,<70,0,-5>,VecCoupe=0,0,1,VecDir=1,0,0
      VecInit=0,-1,0,FinVec=0,-1,0,ÉPAISSEUR=0
      FILTRE/FILTRENUL,
      MODE EXEC=RÉAPPRENDRE
      LIMITE/PLAN,<70,0,-5>,VecPlan=-1,0,0,Croisements=2
      TYPEPALPAGE/VECTEUR
      MODE VALNOMS=RECHERCHEVALNOMS,5
      FINSCAN
      ENDMEAS/
```

Pour créer une droite à partir de ce scanning, vous devez utiliser les fonctions LINESEGMENTSTARTINDEX et LINESEGMENTENDINDEX afin d'extraire les données, comme suit :

```
      ASSIGN/LINESTARTINDEX = LINESEGMENTSTARTINDEX("SCN1", 1, 0.4,0.1)
      ASSIGN/LINEENDINDEX = LINESEGMENTENDINDEX("SCN1", 1, 0.4, 0.1)
```

Ceci commande à PC-DMIS d'aller au scan nommé « SCN1 » et, depuis son premier segment de droite, d'extraire les valeurs d'index de début et de fin comprises entre les tolérances définies. Il affecte ensuite ces valeurs d'index aux variables INDEXDÉBUTDROITE et INDEXFINDROITE.

Une fois les valeurs d'index de début et de fin pour le segment de droite affectées aux variables, vous pouvez employer ces variables dans une droite construite, comme ceci :

```
LIN4=FEAT/LINE,RECT,UNBND
      THEO/100.225,0,-5.011,1,0,0
      ACTL/100.225,-0.005,-5.011,1,-0.0000388,0
      CONSTR/LINE,BF,2D,SCN1.HIT[LINESTARTINDEX..LINEENDINDEX],,
      DÉVIATION_SUPPRESSION/DÉSACTIVER,3
```

```
FILTER/OFF,WAVELENGTH=0
```

Remarquez que dans le code en surbrillance de la droite ci-dessus, PC-DMIS utilise les numéros de début et de fin extraits du scanning pour créer l'élément :
`SCN1.HIT[LINESTARTINDEX..LINEENDINDEX]`

Contrainte d'opérande

Vous pouvez forcer des opérandes à changer de type en utilisant des opérateurs de contrainte :

Contrainte d'entier

INT(<Expression>) - Contraint la valeur de l'expression à un type entier

`INT(4)`	Evaluée à 4
`INT(4.5)`	Evaluée à 4
`INT("Hello World")`	Evaluée à 0
`INT("2")`	Evaluée à 2
`INT("2.2")`	Evaluée à 2
`INT("3 Blind Mice")`	Evaluée à 3
`INT("The 3 Blind Mice")`	Evaluée à 0
`INT("3, 4, 5")`	Evaluée à 3
`INT(MPOINT(0, 0, 1))`	Evaluée à la distance entre le point et l'origine, dans ce cas 1
`INT(MPOINT(3, 4, 5))`	Distance évaluée à 7,0711, cette expression est évaluée à 7

Double contrainte

DOUBLE(<Expression>) - Contraint à entrer deux fois la valeur de l'expression

`DOUBLE(4)`	Evaluée à 4,0
`DOUBLE(4,5)`	Evaluée à 4,5
`DOUBLE("Une chaîne")`	Evaluée à 0,0
`DOUBLE("3,5")`	Evaluée à 3,5
`DOUBLE("3,5 pouces")`	Evaluée à 3,5
`DOUBLE("Le cercle a un diamètre`	Evaluée à 0,0

de 3,5 pouces")	
DOUBLE(MPOINT(0,0,1))	Evaluée à 1,0
DOUBLE(MPOINT(3,4,5))	Evaluée à 7,0711

Contrainte de chaîne

STR(<Expression>) - Contraint la valeur de l'expression à un type chaîne.

STR(4)	Evaluée à« 4 »
STR(4.5)	Evaluée à « 4,5 »
STR("Hello World")	Evaluée à « Bonjour le monde »
STR(MPOINT(3,4,5))	Evaluée à « 3, 4, 5 »

Contrainte de point

MPOINT(<Expression1>, <Expression2>, <Expression3>) - Contraint les valeurs d'expressions à un type point après avoir converti chaque expression en type double.

MPOINT(1, 1, 1)	Evalue à 1,0, 1,0, 1,0
MPOINT(1.1, 1.1, 1.1)	Evalue à 1,1, 1,1, 1,1
MPOINT("1", "1", "1")	Evalue à 1,0, 1,0, 1,0
MPOINT(3, 4.5, "5.6")	Evalue à 3,0, 4,5, 5,6
MPOINT(MPOINT(1, 0, 0), MPOINT(0,1,0), MPOINT(3,4,5))	Evalue à 1,0, 1,0, 7,0711

Contrainte d'opérande et expressions de type mixte

L'évaluateur d'expression force automatiquement les variables dans des expressions de type mixte. Si, en raison d'une contrainte automatique, une expression ne donne pas le résultat attendu, celui-ci est parfois obtenu grâce aux opérateurs de contrainte. Ci-après des exemples de contraintes automatiques dans des expressions de type mixte.

"CIR" + 1
Évalue à "CIR1"

"2" + 2
Évalue à 4

"The Value of 2+2 is " + 2 + 2
Évalue à "The Value of 2+2 is 22" (car les expressions sont évaluées de gauche à droite)

"The Value of 2+2 is " + (2 + 2)
Évalue à "The Value of 2+2 is 4"

LINE1.XYZ > 2
Évalue à 1 si la distance du barycentre de LINE1 depuis l'origine est supérieure à 2.

LINE1.XYZ > LINE2.XYZ
Évalue à 1 si le barycentre de LINE1 est plus loin de l'origine que le barycentre de LINE2.

LINE1.XYZ = LINE2.XYZ
Évalue à 1 si les barycentres de LINE1 et LINE2 sont identiques (pas de contrainte dans ce cas).

DOUBLE(LINE1.XYZ) = DOUBLE(LINE2.XYZ)
Évalue à 1 si les barycentres sont à la même distance de l'origine.

11% 3.1
Évalue à 2 (% est l'oéprateur modulo compatible avec des entiers. Il retourne le reste de la division discrète. 11%3 = 2.)

CIRCLE1.HIT [3.2].X
Évalue à la valeur X mesurée du troisième palpage de Circle1. L'argument 3.2 prend automatiquement la forme d'un entier avec la valeur 3.

Expressions d'identification

De nombreuses commandes PC-DMIS utilisent des ID d'éléments comme paramètres. Par exemple, les éléments construits désignent par des ID les éléments à utiliser comme entrées pour l'élément construit. Les expressions d'identification permettent à l'utilisateur de faire référence à une instance donnée d'élément, à un groupe d'éléments du même nom, à une instance d'élément dans un appel de sous-programme ou à un élément dans une routine de mesure externe.

ID de tableau d'éléments

L'ID de tableau d'éléments permet de faire référence à une instance donnée d'éléments ou à un groupe d'instances d'éléments. Par exemple, si l'élément « Cercle1 » se trouve dans une boucle conditionnelle exécutée cinq fois, ces cinq instances du cercle existent à la sortie de la boucle. Pour vous référer à l'une des cinq instances de « Cercle1 », appliquez la syntaxe du tableau d'éléments décrite à la rubrique « Tableaux d'éléments : » où « Cercle1[1] » correspond à la première instance, « Cercle1[2] » à la deuxième instance, etc.

Pour vous référer à un groupe d'instances, utilisez la notation **..**. "Circle1[1..3]" correspond aux instances 1 à 3 de Circle1. "Circle[3..5]" correspond aux instances 3 à 5 de Circle1. "Circle[1..5]" correspond aux instances 1 à 5 de Circle1. Quand vous faites référence à un groupe d'éléments, celui-ci est traité et se comporte comme une série construite.

Caractères génériques d'ID

Les caractères génériques d'ID font référence à une série d'éléments ayant le même nom. Les deux caractères génériques utilisables sont "*****" et "**?**". (Pour plus d'informations, voir "Sélection d'éléments par mise en correspondance de métacaractères" au chapitre "Modification de l'affichage CAO".)

L'astérisque * permet de faire référence à aucune ou à plusieurs instances d'un caractère quelconque. Pour vous référer à la série de tous les éléments commençant par les lettres "CER", utilisez l'ID d'expression "CER*". Cette syntaxe permet de créer une série qui regroupe tous les éléments dont les ID commencent par "CER", tels que "CERCLE1", "CERCLE2", "CER" ou "CER".

Remarque : si CIR3 a été exécuté plusieurs fois, seule la mesure la plus récente est utilisée. Pour obtenir les différentes instances des exécutions, vous pouvez utiliser l'expression suivante : CIR?[1..3]

Le point d'interrogation '?' est utilisé pour se référer à une seule instance de n'importe quel caractère.

Exemple : l'expression d'identification "MY???1" crée une série d'éléments composés de six caractères, commençant par "MY" et se terminant par "1", tels que "MYCER1", "MYCON1", "MYLIN1" ou "MYFT21".

ID d'éléments dans des sous-programmes, des scripts BASIC ou des routines externes

Les sous-programmes peuvent se trouver dans la routine de mesure actuelle ou dans une routine de mesure externe. Si le sous-programme se trouve dans la même routine que l'appel de sous-programme, vous pouvez utiliser la syntaxe de l'ID de tableau d'élément expliquée dans la rubrique "Tableaux d'éléments" pour faire référence à chaque instance d'élément créé dans ce sous-programme. En revanche, si le sous-programme se trouve dans une routine de mesure externe, vous pouvez utiliser la syntaxe suivante pour faire référence aux éléments créés dans ce sous-programme : "<ID Appel SP>:<ID élément>". Par exemple, si l'élément "F1" figure dans un sous-programme externe appelé depuis une commande Appel SP dont l'ID est "CS1", utilisez l'expression d'identification "CS1:F1" pour vous référer à cet élément.

Exemple : cet exemple n'a d'autre fin que d'illustrer l'application de la syntaxe CS1.F1 et n'est pas destiné à être utilisé.

Routine 1 : PLUS1.PRG

```
SUBROUTINE/PLUS1, A1 = 0, A2 = 0, A3 = 0
F1 =FEAT/POINT,RECT
THEO /A1+1,A2+1,A3+1,0,0,1
ACTL/3,1,1,0,0,1
```

```
MEAS/POINT,1
HIT /BASIC,A1+1,A2+1,A3+1,0,0,1,0,0,0
ENDMEAS/
ENDSUB/
```

Routine 2 : TEST.PRG

```
CS1 =CALLSUB/PLUS1,D:\V30\WINDEBUG\PLUS1.PRG: 3,3,3,,
DIM D1= LOCATION OF POINT CS1:F1 UNITS=IN,$
GRAPH=OFF TEXT=OFF MULT=10.00 OUTPUT=BOTH
AX NOMINAL +TOL -TOL MEAS MAX MIN DEV OUTTOL
X 3.0000 0.0000 0.0000 3.0000 3.0000 3.0000 0.0000 0.0000
----#----
END OF DIMENSION D1
```

Les scripts BASIC créent et suppriment les objets de manière dynamique. Appliquez la syntaxe « <ID script Basic>:<ID élément> » pour faire référence à un élément créé par un script BASIC. Par exemple, si un script BASIC dont l'ID est « BS1 » crée l'élément « F2 », utilisez l'expression d'identification « BS1:F2 » pour vous référer à cet élément.

La commande ATTACH permet de joindre des routines externes à PC-DMIS. Pour faire référence à des éléments contenus dans la routine jointe, utilisez la syntaxe suivante : "<Attach ID routine>:<ID élément>". Pour faire référence à l'élément "F3" dans la routine de mesure jointe "GEAR1", utilisez l'expression "GEAR1:F3". (Pour plus d'informations, voir "Attachement d'une routine de mesure externe" au chapitre "Ajout d'éléments externes").

Combinaisons d'expressions d'identification

Des expressions d'identification de tableaux , d'identification par caractères génériques, d'identification de sous-programmes et de scripts BASIC et de routines de mesure externes peuvent être combinées. Exemple : pour faire référence à la troisième instance de tous les éléments commençant par "CIR" dans une routine de mesure externe jointe avec l'ID "BOLTPAT", utilisez l'expression d'identification "`BOLTPAT:CER*[3]`".

Il est aussi possible d'utiliser les expressions d'identification comme des expressionsnormales. Vous pouvez ainsi affecter le barycentre mesuré de la série d'éléments précédente à une variable en utilisant l'expression suivante :

```
ASSIGN/V1 = BOLTPAT:CIR*[3].XYZ
```

Il est aussi possible d'utiliser les expressions d'identification comme des expressionsnormales. Vous pouvez ainsi affecter le barycentre mesuré de la série d'éléments précédente à une variable en utilisant l'expression suivante :

```
ASSIGN/V1 = BOLTPAT:CIR*[3].XYZ
```

Accès aux propriétés d'un objet de rapport

Vous pouvez créer vos propres modèles de rapport et d'étiquette. PC-DMIS s'en sert pour afficher les données de rapport dans une fenêtre Rapport (voir **Afficher | Fenêtre de rapport**). Les modèles sont créés à l'aide d'éditeurs qui utilisent une interface de type Visual Basic dans laquelle insérer, déplacer et redimensionner des composants spéciaux appelés "objets".

Chaque objet possède des "propriétés" qui définissent son mode d'affichage et les informations qu'il renferme. Certaines propriétés sont communes à tous les objets, d'autres seulement aux objets en relation, et d'autres encore sont uniques à un objet spécifique.

Le langage d'expression de PC-DMIS peut appeler le rapport actuellement chargé et stocker les valeurs de propriétés d'un objet déterminé dans une variable. Il peut obtenir des valeurs de type Chaîne, Nombre entier et Réel grâce à l'aide de cette syntaxe :

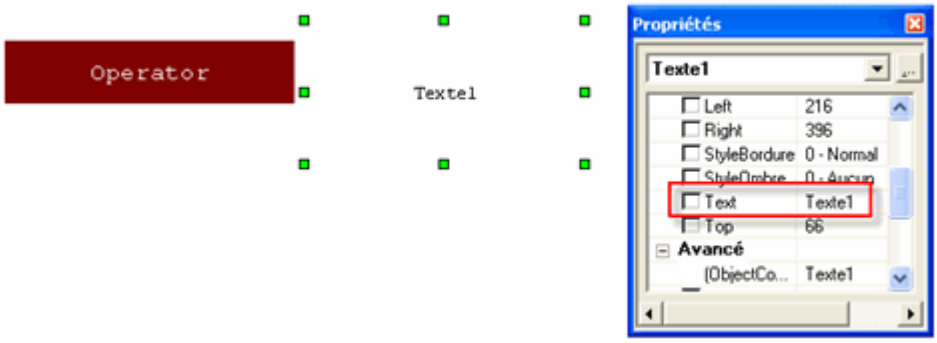
Syntaxe de demande de propriété

```
Assign/V1 = Rapport.<nom de l'objet>.<nom de la propriété>
```

"Report" fait référence au rapport actuellement chargé. <nom de l'objet> correspond au nom unique de l'objet et <nom de la propriété> est un nom de propriété valide pour cet objet.

Exemple

Imaginez que votre modèle de rapport comporte un objet texte nommé "Texte1" que vous souhaitez utiliser dans le rapport final pour afficher le nom de l'opérateur. La chaîne de caractères représentant le nom de l'opérateur sera stockée dans la propriété **Text** de l'objet. Par défaut, cette propriété (texte affiché) possède à l'origine la valeur "Texte1" (voir l'illustration ci-dessous). Comme il s'agit d'une propriété affectée par l'utilisateur, elle change lorsque vous entrez le nom au cours de l'exécution.



Boîte de dialogue Propriétés montrant un objet sélectionné et la propriété à demander

Pour utiliser le code du langage d'expression afin de demander la propriété « Texte » de cet objet texte et obtenir les données saisies, utilisez la commande suivante :

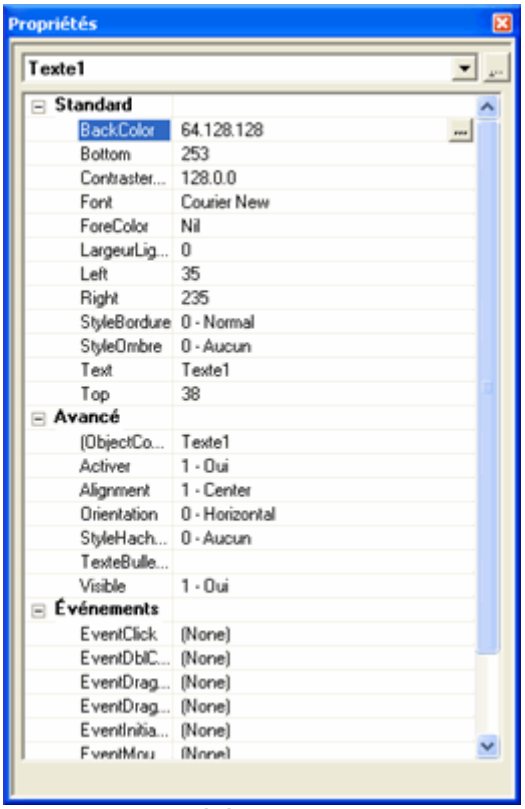
```
Assign/V1 = Report.Text1.Text
```

Dans ce code :

- Dans ce code, "Report" commande d'analyser le rapport chargé dans la fenêtre de rapport.
- « Texte1 » lui indique qu'il faut rechercher l'objet « Texte1 ».
- "Text" indique qu'il faut rechercher la propriété "Text" dans cet objet. La valeur de la propriété "Text" est ensuite transmise à la variable V1, que vous pouvez alors manipuler ou afficher à l'aide du langage d'expression de PC-DMIS.

Recherche de propriétés

Vous pouvez identifier les propriétés associées à un objet donné en accédant au modèle de rapport dans l'éditeur de modèle de rapport (**Fichier | Gén rapports | Modifier | Modèle rapport**), en sélectionnant l'objet et en cliquant dessus avec le bouton droit pour afficher sa feuille de propriétés.



Feuille de propriétés d'un objet texte

La feuille de propriétés contient deux colonnes. La colonne de gauche affiche le nom de la propriété, celle de droite la valeur actuelle. Veillez à utiliser le nom de propriété exact dans votre code d'expression.

Important : lors de la demande de valeurs de propriétés, certaines propriétés peuvent renvoyer une valeur numérique apparemment inutile. Tel est généralement le cas lorsque la propriété

possède une liste définie d'options disponibles et que PC-DMIS renvoie pour la propriété sélectionnée une valeur interne sans relation avec la propriété affichée.

Par exemple, l'objet **Texte** possède une propriété **Orientation** comportant ces valeurs :
0 - Horizontal
1 - Vertical haut
2 - Vertical bas

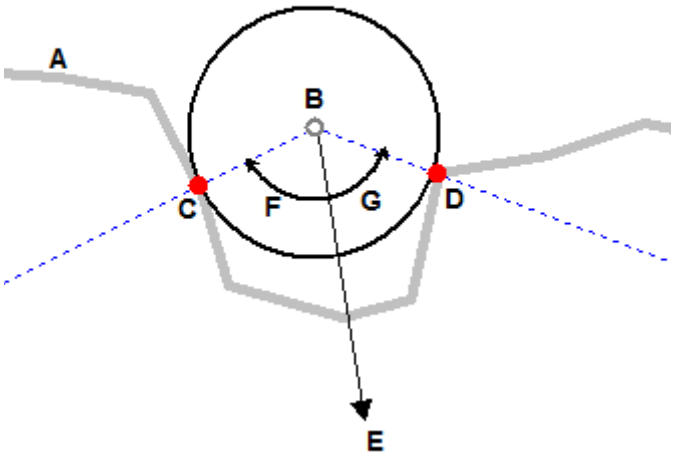
Cependant, si vous obtenez la valeur à l'aide du langage d'expression PC-DMIS, le logiciel renverra plutôt ce qui suit :
0 (pour Horizontal)
900 (pour Vertical haut)
-900 (pour Vertical bas)

Cela peut demander plusieurs essais et erreurs avant de déterminer quelles valeurs de renvoi correspondent à la valeur affichée sur la feuille de propriétés.

Accès aux informations à partir d'un cercle minimum de scan construit

En utilisant des expressions PC-DMIS vous pouvez tirer des informations d'un élément de cercle construit avec un rayon donné à un point minimum le long d'un scan linéaire. Voir la rubrique "Construction d'un cercle à un point minimum de scan au chapitre "Construction de nouveaux éléments à partir d'éléments existants" pour plus d'informations.

Lorsque vous construisez un élément de cercle minimum de scan, le cercle utilise ultimement un vecteur (appelé vecteur vers le bas) pour entrer en contact avec la ligne de scan. Il entre en contact avec la ligne seulement en deux endroits appelés points de contact (POINTDECONTACT1 et POINTDECONTACT2). PC-DMIS peut ensuite utiliser ces points pour déterminer les angles à partir du vecteur vers le bas vers ces points de contact (ANGLEDECONTACT1 et ANGLEDECONTACT2). Par exemple, considérez ce diagramme :



A - La ligne de scan à laquelle le cercle est construit.
B - La position finale XYZ du barycentre du cercle.
C - Le point de contact à gauche du vecteur vers le bas. Il est appelé CONTACTPOINT1.
D - Le point de contact à droite du vecteur vers le bas. Il est appelé CONTACTPOINT2.
E - Vecteur vers le bas.
F - Retourne l'angle entre le vecteur vers le bas et le CONTACTPOINT1. Il est appelé CONTACTANGLE1.
G - Retourne l'angle entre le vecteur vers le bas et le CONTACTPOINT2. Il est appelé CONTACTANGLE2.

Les expressions détaillées ci-dessous ne fonctionnent qu'avec ce type d'élément de cercle construit. Vous pouvez aussi utiliser CONTACTPOINT2 dans la syntaxe ci-dessous pour retourner les informations équivalentes utilisant le deuxième point de contact.

`ASSIGN/V1=CIR1.CONTACTPOINT1. XYZ`
Retourne les informations de point XYZ pour le premier point de contact du cercle avec la ligne CONTACTPOINT1.

`ASSIGN/V1=CIR1.CONTACTPOINT1.X`
Retourne les informations X pour CONTACTPOINT1.

`ASSIGN/V1=CIR1.CONTACTPOINT1.Y`
Retourne les informations Y pour CONTACTPOINT1.

`ASSIGN/V1=CIR1.CONTACTPOINT1.Z`
Retourne les informations Z pour CONTACTPOINT1.

`ASSIGN/V1=CIR1.CONTACTPOINT1.IJK`
Retourne le vecteur IJK depuis CONTACTPOINT1 vers le barycentre du cercle.

`ASSIGN/V1=CIR1.CONTACTPOINT1.I`
Retourne la valeur I depuis le vecteur CONTACTPOINT1 IJK ci-dessus.

`ASSIGN/V1=CIR1.CONTACTPOINT1.J`
Retourne la valeur J depuis le vecteur CONTACTPOINT1 IJK ci-dessus.

`ASSIGN/V1=CIR1.CONTACTPOINT1.K`
Retourne la valeur K depuis le vecteur CONTACTPOINT1 IJK ci-dessus.

`ASSIGN/V1=CIR1.CONTACTANGLE1`
Retourne l'angle depuis le vecteur du bas vers CONTACTPOINT1.

`ASSIGN/V1=CIR1.CONTACTANGLE2`
Retourne l'angle depuis le vecteur du bas vers CONTACTPOINT2.

`ASSIGN/V1=CIR1.CONTACTANGLE`
Retourne la somme des valeurs absolues de CONTACTANGLE1 et CONTACTANGLE2. Elle ne doit pas dépasser 180 degrés.