7.1.1.5 Reset Service

The service procedure of the Reset service shall be extended.

The MMS service procedure, with the extensions specified in this companion standard, shall be repeated for all Program Invocations which have the attribute Independent equal to FALSE, and the attribute ProgramInvocationReference equal to the name of the reset Program Invocation. The results of trying to reset each of these Program Invocations shall be ignored.

7.1.1.6 Kill Service

The service procedure of the Kill service shall be extended.

The I/O State attribute of the Program Invocation shall be set according to the I/O State parameter of the CS-Kill-Request argument. The Interface Function to Sensors and Actuators shall be directed as specified in the following table:

I/O State	Actuator Interface
Hold Current State	Hold current values
Implementer State	Set all outputs controlled by this application program according to the implementers specification, then hold current values
Zero Outputs	Set all outputs controlled by this application program to zero, then hold current values
User Specified	Set the specified outputs controlled by this application program to the user specified values, then hold current values

If the I/O State parameter is omitted, the default (implementer state) shall be used.

The MMS service procedure, with the extensions specified in this companion standard, shall be repeated for all Program Invocations which have the attribute Independent equal to FALSE, and the attribute ProgramInvocationReference equal to the name of the killed Program Invocation. The value for the I/O State parameter to be used for killing each of these Program Invocations shall be the same as that used to kill this Program Invocation. The results of trying to kill each of these Program Invocations shall be ignored.

The structure of the CS-Kill-Request argument is shown in table 6.

Table 6 – CS-Kill-Request argument

Parameter Name	Req	Ind
I/O State	U	U(=)

7.1.1.7 GetProgramInvocationAttributes Service

The service procedure of the GetProgramInvocationAttributes service shall be extended.

The I/O State attribute is reported as a parameter in the CS-GetProgramInvocationAttributes-Response argument. If the attribute Independent has a value of FALSE, the Program Invocation Reference attribute is also reported as a parameter in the CS-GetProgram InvocationAttributes-Response argument.

The structure of the CS-GetProgramInvocationAttributes-Response argument is shown in table 7.

Tableau 7 – Argument CS-GetProgramInvocationAttributes-Response

Parameter Name	Rsp	Cnf
I/O State	M	M(=)
Program Invocation Reference	C	C(=)

7.1.1.8 Service CreateProgramInvocation

La procédure du service CreateProgramInvocation doit être élargie.

L'attribut I/O State est initialisé à ImplementerState. L'interface vers les actionneurs doit définir toutes les sorties contrôlées par ce programme d'application en fonction de la spécification des concepteurs, puis garder les valeurs en cours.

Si le paramètre Program Invocation Reference est présent, l'attribut Program Invocation Reference doit avoir la valeur du paramètre Program Invocation Reference et l'attribut Independent doit être égal à FAUX. Si l'invocation de programme référencée a l'attribut Independent FAUX et que la mise en oeuvre ne le permet pas, une erreur «Définition, Autre" doit être renvoyée.

Si le paramètre Program Invocation Reference n'est pas présent, l'attribut Independent doit être VRAI.

La structure de l'argument CS-CreateProgramInvocation-Request est définie dans le tableau 8.

Tableau 8 – Argument CS-CreateProgramInvocation-Request

Parameter Name	Req	Ind
Program Invocation Reference	U	U(=)

7.1.2 Additions à la définition des protocoles MMS

L'ISO-9506-MMS-CSPC-1 représente la version numéro 1 de la norme d'accompagnement pour les Automates Programmables.

NOTE – Certains termes utilisés dans les paragraphes ci-dessous ne sont pas définis dans cette partie de la norme. Ils sont répertoriés dans la section «Imports" suivante et sont définis dans l'ISO/CEI 9506-2.

```
-- ------------------------------------------------------------------
ISO-9506-MMS-CSPC-1 {iso standard 9506 part (5) mms-cspc-version1 (1)}
DEFINITIONS::= BEGIN
  IMPORTS                                   MMSpdu,
                                                 ParameterSupportOptions,
                                                 ServiceSupportOptions,
                                                 Integer16,
                                                 StatusResponse,
                                                 Identifier,
                                                 Data,
                                                 ObjectName

  FROM MMS-General-Module-1                      {iso standard 9506 part

(2) mms-general-module-version1 (2)};
InitRequestDetail::= SEQUENCE {
                                          proposedVersionNumber        [0]
IMPLICIT Integer16,
                                          proposedParameterCBB         [1]
IMPLICIT

  ParameterSupportOptions
                                          servicesSupportedCalling     [2]
IMPLICIT
                                                   ServiceSupportOptions
                                          }
```

Table 7 – CS-GetProgramInvocationAttributes-Response argument

Parameter Name	Rsp	Cnf
I/O State	M	M(=)
Program Invocation Reference	C	C(=)

7.1.1.8 CreateProgramInvocation Service

The service procedure of the CreateProgramInvocation service shall be extended.

The I/O State attribute is set to implementerState. Direct the Interface to Actuators to set all outputs controlled by this application program according the the implementers specification, then hold current values.

If the Program Invocation Reference parameter is present, then the Program Invocation Reference attribute is set to the value of the Program Invocation Reference parameter and the Independent attribute is set to FALSE. If the referenced Program Invocation has the Independent attribute set to FALSE and the implementation does not permit this, then an error of "Definition, Other" shall be returned.

If the Program Invocation Reference parameter is not present, then the Independent attribute is set to TRUE.

The structure of the CS-CreateProgramInvocation-Request argument is shown in table 8.

Table 8 – CS-CreateProgramInvocation-Request argument

Parameter Name	Req	Ind
Program Invocation Reference	U	U(=)

7.1.2 Additions to the MMS Protocol Definition

ISO-9506-MMS-CSPC-1 represents version number 1 of the MMS Companion Standard for Programmable Controllers.

NOTE – There are some terms that are used in the following productions which are not defined in this part of this standard. These are listed in the "IMPORTS" section of what follows and are defined in ISO/IEC 9506-2.

```
ISO-9506-MMS-CSPC-1 {iso standard 9506 part(5) mms-cspc-version1(1)}
DEFINITIONS ::= BEGIN
   IMPORTS  MMSpdu,
            ParameterSupportOptions,
            ServiceSupportOptions,
            Integer16,
            StatusResponse,
            Identifier,
            Data,
            ObjectName
   FROM MMS-General-Module-1
            {iso standard 9506 part(2) mms-general-module-version1(2)};
InitRequestDetail ::= SEQUENCE {
         proposedVersionNumber       [0] IMPLICIT Integer16,
         proposedParameterCBB        [1] IMPLICIT
                                  ParameterSupportOptions,
         servicesSupportedCalling    [2] IMPLICIT
                                  ServiceSupportOptions
         }
```

```
InitResponseDetail::= SEQUENCE {
  negociatedVersionNumber              [0]   IMPLICIT Integer16,
  negociatedParameterCBB               [1]   IMPLICIT
                                             ParameterSupportOptions,
  sercicesSupportedCalled              [2]   IMPLICIT
                                             ServiceSupportOptions
  }
CS-Status-Request::= NULL
CS-GetNameList-Request::= NULL
CS-Input-Request::= NULL
CS-Output-Request::= NULL
CS-InitiateDownloadSequence-Request::=NULL
CS-DownloadSegment-Request::= NULL
CS-TerminateDownloadSequence-Request::= NULL
CS-InitiateUploadSequence-Request::= NULL
CS-UploadSegment-Request::= NULL
CS-TerminateUploadSequence-Request::= NULL
CS-RequestDomainDownload-Request::= NULL
CS-RequestDomainUpload-Request::= NULL
CS-LoadDomainContent-Request::= NULL
CS-StoreDomainContent-Request::= NULL
CS-DeleteDomain-Request::= NULL
CS-GetDomainAttributes-Request::= NULL
CS-CreateProgramInvocation-Request::= Identifier
                                      -- Nothing shall be transmitted
if no identifier is given
CS-DeleteProgramInvocation-Request::= NULL
IoState::= INTEGER {
  controlled       (0),
  holdOutputs      (1),
  holdCurrentState (2),
  implementerState (3),
  zeroOutputs      (4),
  userSpecified           (5)
  }
CS-Start-Request::= OPTIONAL IoState -- Only values 0,1,2 are valid;
default is 0
CS-Stop-Request::= OPTIONAL IoState -- Only values 2,3,4,5 are valid;
default is 3
CS-Resume-Request::= OPTIONAL IoState -- Only values 0,1,2 are valid;
default is 0
CS-Reset-Request::= NULL
CS-Kill-Request::= OPTIONAL IoState -- Only values 2,3,4,5 are valid;
default is 3
CS-GetProgramInvocationAttributes-Request::= NULL
CS-DefineEventcondition-Request::= NULL
CS-DeleteEventCondition-Request::= NULL
CS-GetEventConditionAttributes-Request::= NULL
CS-ReportEventConditionStatus-Request::= NULL
CS-AlterEventConditionMonitoring-Request::= NULL
CS-TriggerEvent-Request::= NULL
CS-DefineEventAction-Request::= NULL
CS-DeleteEventAction-Request::= NULL
CS-GetEventActionAttributes-Request::= NULL
CS-ReportEventActionStatus-Request::= NULL
CS-DefineEventEnrollment-Request::= NULL
CS-DeleteEventEnrollment-Request::= NULL
CS-AlterEventenrollment-Request::= NULL
CS-ReportEventEnrollmentStatus-Request::= NULL
CS-GetEventEnrollmentAttributes-Request::= NULL
CS-AchnowledgeEventNotification-Request::= NULL
CS-GetAlarmSummary-Request::= NULL
CS-GetAlarmEnrollmentSummary-Request::=NULL
CS-ReadJournal-Request::= NULL
CS-WriteJournal-Request::= NULL
CS-InitializeJournal-Request::= NULL
CS-ReportJournalStatus-Request::= NULL
```

```
InitResponseDetail ::= SEQUENCE {
        negotiatedVersionNumber     [0] IMPLICIT Integer16,
        negotiatedParameterCBB      [1] IMPLICIT
                                ParameterSupportOptions,
        servicesSupportedCalled     [2] IMPLICIT
                                ServiceSupportOptions
        }
CS-Status-Request ::= NULL
CS-GetNameList-Request ::= NULL
CS-Input-Request ::= NULL
CS-Output-Request ::= NULL
CS-InitiateDownloadSequence-Request ::= NULL
CS-DownloadSegment-Request ::= NULL
CS-TerminateDownloadSequence-Request ::= NULL
CS-InitiateUploadSequence-Request ::= NULL
CS-UploadSegment-Request ::= NULL
CS-TerminateUploadSequence-Request ::= NULL
CS-RequestDomainDownload-Request ::= NULL
CS-RequestDomainUpload-Request ::= NULL
CS-LoadDomainContent-Request ::= NULL
CS-StoreDomainContent-Request ::= NULL
CS-DeleteDomain-Request ::= NULL
CS-GetDomainAttributes-Request ::= NULL
CS-CreateProgramInvocation-Request ::= Identifier
              -- Nothing shall be transmitted if no identifier is
given
CS-DeleteProgramInvocation-Request ::= NULL
IoState ::= INTEGER  {
controlled            (0),
holdOutputs           (1),
holdCurrentState      (2),
implementerState      (3),
zeroOutputs           (4),
userSpecified         (5)
      }
CS-Start-Request   ::= OPTIONAL IoState   -- Only values 0, 1, 2 are
valid; default is 0
CS-Stop-Request   ::= OPTIONAL IoState -- Only values 2, 3, 4, 5 are
valid; default is 3
CS-Resume-Request   ::= OPTIONAL IoState   -- Only values 0, 1, 2 are
valid; default is 0
CS-Reset-Request ::= NULL
CS-Kill-Request ::= OPTIONAL IoState   -- Only values 2, 3, 4, 5 are
valid; default is 3
CS-GetProgramInvocationAttributes-Request ::= NULL
CS-DefineEventCondition-Request ::= NULL
CS-DeleteEventCondition-Request ::= NULL
CS-GetEventConditionAttributes-Request ::= NULL
CS-ReportEventConditionStatus-Request ::= NULL
CS-AlterEventConditionMonitoring-Request ::= NULL
CS-TriggerEvent-Request ::= NULL
CS-DefineEventAction-Request ::= NULL
CS-DeleteEventAction-Request ::= NULL
CS-GetEventActionAttributes-Request ::= NULL
CS-ReportEventActionStatus-Request ::= NULL
CS-DefineEventEnrollment-Request ::= NULL
CS-DeleteEventEnrollment-Request ::= NULL
CS-AlterEventEnrollment-Request ::= NULL
CS-ReportEventEnrollmentStatus-Request ::= NULL
CS-GetEventEnrollmentAttributes-Request ::= NULL
CS-AcknowledgeEventNotification-Request ::= NULL
CS-GetAlarmSummary-Request ::= NULL
CS-GetAlarmEnrollmentSummary-Request ::= NULL
CS-ReadJournal-Request ::= NULL
CS-WriteJournal-Request ::= NULL
CS-InitializeJournal-Request ::= NULL
CS-ReportJournalStatus-Request ::= NULL
```

```
CS-CreateJournal-Request::= NULL
CS-DeleteJournal-Request::= NULL
CS-GetCapabilityList-Request::= NULL
CS-Status-Response::= NULL
CS-GetNameList-Response::= NULL
CS-Input-Response::= NULL
CS-Ouput-Response::= NULL
CS-InitiateDownloadSequence-Response::= NULL
CS-DownloadSegment-Response::= NULL
CS-TerminateDownloadSequence-Response::= NULL
CS-InitiateUploadSequence-Response::= NULL
CS-UploadSegment-Response::= NULL
CS-TerminateUploadSequence-Response::= NULL
CS-RequestDomainDownload-Response::= NULL
CS-RequestDomainUpload-Response::= NULL
CS-LoadDomainContent-Response::= NULL
CS-StoreDomainContent-Response::= NULL
CS-DeleteDomain-Response::= NULL
CS-GetDomainAttributes-Response::= NULL
CS-CreateProgramInvocation-Response::= NULL
CS-DeleteProgramInvocation-Response::= NULL
CS-Start-Response::= NULL
CS-Stop-Response::= NULL
CS-Resume-Response::= NULL
CS-Reset-Response::= NULL
CS-Kill-Response::= NULL
CS-GetProgramInvocationAttributes-Response::= SEQUENCE {
  ioState                                [0] IoState,
  programInvocationReference [1] IMPLICIT Identifier OPTIONAL
   -- Only provided if value of Independent attribute is FALSE
  }
CS-DefineEventCondition-Response::= NULL
CS-DeleteEventCondition-Response::= NULL
CS-GetEventConditionAttributes-Response::= NULL
CS-ReportEventConditionStatus-Response::= NULL
CS-AlterEventConditionMonitoring-Response::= NULL
CS-TriggerEvent-Response::= NULL
CS-DefineEventAction-Response::= NULL
CS-DeleteEventAction-Response::= NULL
CS-GetEventActionAttributes-Response::= NULL
CS-ReportEventActionStatus-Response::= NULL
CS-DefineEventEnrollment-Response::= NULL
CS-DeleteEventEnrollment-Response::= NULL
CS-AlterEventEnrollment-Response::= NULL
CS-ReportEventEnrollmentStatus-Response::= NULL
CS-GetEventEnrollmentAttributes-Response::= NULL
CS-AcknowledgeEventNotification-Response::= NULL
CS-GetAlarmSummary-Response::= NULL
CS-GetAlarmEnrollmentSummary-Response::= NULL
CS-ReadJournal-Response::= NULL
CS-WriteJournal-Response::= NULL
CS-InitializeJournal-Response::= NULL
CS-ReportJournalStatus-Response::= NULL
CS-CreateJournal-Response::= NULL
CS-DeleteJournal-Response::= NULL
CS-GetCapabilityList-Response::= NULL
CS-UnsolicitedStatus::= NULL
CS-EventNotification::= NULL
CS-EventNotification::= NULL
CS-Service-Error::= NULL
CS-GetDataExchangeAttributes-Request::= NULL
CS-GetDataExchangeAttributes-Response::= NULL
CS-ExchangeData-Request::= NULL
CS-ExchangeData-Response::= NULL
AdditionalService-Error::= NULL
AdditionalUnconfirmedService::= NULL
CsAdditionalObjectClasses::= NULL
```

```
CS-CreateJournal-Request ::= NULL
CS-DeleteJournal-Request ::= NULL
CS-GetCapabilityList-Request ::= NULL
CS-Status-Response ::= NULL
CS-GetNameList-Response ::= NULL
CS-Input-Response ::= NULL
CS-Output-Response ::= NULL
CS-InitiateDownloadSequence-Response ::= NULL
CS-DownloadSegment-Response ::= NULL
CS-TerminateDownloadSequence-Response ::= NULL
CS-InitiateUploadSequence-Response  ::= NULL
CS-UploadSegment-Response ::= NULL
CS-TerminateUploadSequence-Response ::= NULL
CS-RequestDomainDownload-Response ::= NULL
CS-RequestDomainUpload-Response ::= NULL
CS-LoadDomainContent-Response ::= NULL
CS-StoreDomainContent-Response ::= NULL
CS-DeleteDomain-Response ::= NULL
CS-GetDomainAttributes-Response ::= NULL
CS-CreateProgramInvocation-Response ::= NULL
CS-DeleteProgramInvocation-Response ::= NULL
CS-Start-Response ::= NULL
CS-Stop-Response ::= NULL
CS-Resume-Response ::= NULL
CS-Reset-Response ::= NULL
CS-Kill-Response ::= NULL
CS-GetProgramInvocationAttributes-Response ::= SEQUENCE {
   ioState                      [0] IoState,
   programInvocationReference[1] IMPLICIT Identifier OPTIONAL
      -- Only provided if value of Independent attribute is FALSE
   }
CS-DefineEventCondition-Response ::= NULL
CS-DeleteEventCondition-Response ::= NULL
CS-GetEventConditionAttributes-Response ::= NULL
CS-ReportEventConditionStatus-Response ::= NULL
CS-AlterEventConditionMonitoring-Response ::= NULL
CS-TriggerEvent-Response ::= NULL
CS-DefineEventAction-Response ::= NULL
CS-DeleteEventAction-Response ::= NULL
CS-GetEventActionAttributes-Response ::= NULL
CS-ReportEventActionStatus-Response ::= NULL
CS-DefineEventEnrollment-Response ::= NULL
CS-DeleteEventEnrollment-Response ::= NULL
CS-AlterEventEnrollment-Response ::= NULL
CS-ReportEventEnrollmentStatus-Response ::= NULL
CS-GetEventEnrollmentAttributes-Response ::= NULL
CS-AcknowledgeEventNotification-Response ::= NULL
CS-GetAlarmSummary-Response ::= NULL
CS-GetAlarmEnrollmentSummary-Response ::= NULL
CS-ReadJournal-Response ::= NULL
CS-WriteJournal-Response ::= NULL
CS-InitializeJournal-Response ::= NULL
CS-ReportJournalStatus-Response ::= NULL
CS-CreateJournal-Response ::= NULL
CS-DeleteJournal-Response ::= NULL
CS-GetCapabilityList-Response ::= NULL
CS-UnsolicitedStatus ::= NULL
CS-EventNotification ::= NULL
CS-Service-Error ::= NULL
CS-GetDataExchangeAttributes-Request ::= NULL
CS-GetDataExchangeAttributes-Response ::= NULL
CS-ExchangeData-Request ::= NULL
CS-ExchangeData-Response ::= NULL
AdditionalService-Error ::= NULL
AdditionalUnconfirmedService ::= NULL
CsAdditionalObjectClasses ::= NULL
```

```
EN-Additional-Detail::= NULL
EE-Additional-Detail::= NULL
JOU-Additional-Detail::= NULL
--------------------------------------------------------------------------
```

8 Noms normalisés

Cet article précise les noms normalisés définis dans cette norme.

8.1 Objets Domaines – AUCUN

8.2 Objets Invocation de Programme – AUCUN

8.3 Objets Variable Nommée

Il existe trois objets Variable Nommée normalisés:

a) P_DDATE

b) P_PCSTATE

c) P_PCSTATUS

P_DDATE est une variable spécifique au domaine. Elle contient la date de la modification la plus récente du programme d'application qui est contenu dans le domaine. Elle est générée au niveau local.

P_PCSTATE est une chaîne binaire qui contient l'information sur l'état de santé et la valeur d'état du PC. Le paragraphe 5.1.4. contient une description sémantique de l'état et de la santé du PC. Cette variable est fournie aux clients qui souhaitent obtenir l'état en même temps que les données. La méthodologie pour maintenir cette variable à jour doit être précisée par le concepteur. Les méthodologies possibles comprennent ce qui suit:

a) la périodicité;

b) dès que la variable est lue;

c) une fois par temps de cycle de programme.

P_PCSTATUS est une variable structurée qui contient l'information relative à la santé et à l'état du PC et de ses sous-systèmes. Cette variable est fournie aux clients qui souhaitent obtenir des informations complémentaires sur l'état. La méthode pour maintenir à jour cette variable doit être définie par le concepteur dans la PICS.

Les définitions formelles de chacune de ces variables sont données ci-après.

8.3.1 P_DDATE

Un ensemble d'objets Variable Nommée spécifiques au domaine doit être défini pour rassembler l'information en provenance de domaines existants dans le VMD.

```
Object: Named Variable
Key Attribute: Variable Name = Domain-specific {
                 domainID,
                 itemID   "P_DDATE" }
     Attribute: Reference to Access Control List = 'M_ReadOnly'
     Attribute: Type Description = binary-time TRUE
                 -- contains date and time
     Attribute: Access Method = non-public
```

```
EN-Additional-Detail ::= NULL
EE-Additional-Detail ::= NULL
JOU-Additional-Detail ::= NULL
```

---

8 Standardized names

This clause specifies the standardized names defined in this standard.

8.1 Domain Objects – NONE

8.2 Program Invocation Objects – NONE

8.3 Named Variable Objects

There are three standardized named variable objects:

a) P_DDATE

b) P_PCSTATE

c) P_PCSTATUS

P_DDATE is a domain specific variable. It contains the date of the most recent modification of the application program that is contained in the domain. It is generated by local means.

P_PCSTATE is a bit string that contains health and state information about the PC. Subclause 5.1.4 contains a semantic description of the PC state and health. This variable is provided for those clients that wish to obtain status at the same time data is acquired. The methodology of maintaining this variable is to be specified by the implementer. Some possible methodologies include the following:

a) Periodic time basis,

b) Whenever this variable is read,

c) Once per program scan.

P_PCSTATUS is a structured variable that contains health and status information about the PC and its subsystems. This variable is provided for those clients which wish to obtain more detailed status information. The method of maintaining this variable shall be specified by the implementer in the PICS.

The formal definitions of each of these follows.

8.3.1 P_DDATE

A set of domain-specific Named Variable objects shall be defined to achieve information from existing Domains in the VMD.

```
Object: Named Variable
Key Attribute: Variable Name = domain-specific {
                  domainID ,
                  itemID   "P_DDATE" }
    Attribute: Reference to Access Control List = 'M_ReadOnly'
    Attribute: Type Description = binary-time TRUE
                  -- contains date and time
    Attribute: Access Method = non-public
```

8.3.2 P_PCSTATE

Un objet Variable Nommée spécifique au VMD doit contenir le statut récapitulatif du PC.

```
Object: Named Variable
Key Attribute: Variable Name = vmd-specific "P_PCSTATE"
    Attribute: Reference to Access Control List = 'M_ReadOnly'
    Attribute: Type Description = bit-string 16
--      good              (0),
--      warning           (1),
--      bad               (2),
--      running           (3),
--      localControl      (4),
--      noOutputsDisabled  (5),
--      noInputsDisabled (6),
--      forced            (7),
--      appPresent        (8),
--      ioFault           (9),
--      puFault           (10),
--      powFault          (11),
--      memFault          (12),
--      comFault          (13),
--      implementerFault (14)
    Attribute: Access Method = non-public
```

La sémantique des bits de la variable P_PCSTATE résulte de l'information sur l'état de santé et les valeurs d'état des sous-systèmes du PC (voir 5.1.4):

– good – S'il est VRAI, cet attribut indique que le système PC et tous ses sous-systèmes sont en état de santé. Cet attribut est VRAI si et seulement si tous les sous-systèmes du PC indiquent un état de santé GOOD;

– warning – S'il est VRAI, cet attribut indique qu'au moins un sous-système du PC indique un état de santé WARNING et qu'aucun d'entre eux n'a un état de santé BAD;

– bad – S'il est VRAI, cet attribut indique qu'au moins un sous-système indique un état de santé BAD;

– running – S'il est VRAI, cet attribut indique qu'au moins une partie de l'application utilisateur a été chargée et qu'elle est sous le contrôle du PC;

– localControl – S'il est VRAI, cet attribut indique que la commande locale de substitution est active. Si elle l'est, la possibilité de contrôler le PC et ses sous-systèmes à partir du réseau peut être limitée. Cela peut par exemple être fortement lié à l'utilisation d'une clé de commutation locale;

– noOutputsDisabled – S'il est VRAI, cet attribut indique que le PC peut modifier l'état physique de toutes les sorties en exécutant le programme d'application ou par d'autres moyens. S'il n'est pas VRAI, l'état physique de certaines sorties n'est pas modifiable (leur état logique pouvant l'être). Cet attribut est généralement utilisé lors de la mise au point des programmes d'application des programmes d'application dans le PC;

– noInputsDisabled – S'il est VRAI, cet attribut indique que le PC peut accéder à l'état physique de toutes les entrées en exécutant le programme d'application ou par un d'autres moyens. S'il n'est pas VRAI, il n'est pas possible d'accéder à l'état physique de certaines entrées. Cet attribut est généralement utilisé lors de la mise au point des programmes d'application dans lesquels les entrées peuvent être simulées;

– forced – S'il est VRAI, cet attribut indique qu'un moins un point d'entrées-sorties ou une variable associé(e) au PC a été forcé(e). Lorsqu'une variable est forcée, le programme d'application doit utiliser la valeur spécifiée par le PADT et non la valeur provenant de l'exécution normale du programme;

– appPresent – S'il est VRAI, cet attribut indique qu'il existe au moins une application utilisateur dans le PC;

8.3.2 P_PCSTATE

A VMD-specific Named Variable object shall contain the summary status of the PC.

```
Object: Named Variable
Key Attribute: Variable Name = vmd-specific "P_PCSTATE"
    Attribute: Reference to Access Control List = 'M_ReadOnly'
    Attribute: Type Description = bit-string 16
--      good                (0),
--      warning             (1),
--      bad                 (2),
--      running             (3),
--      localControl        (4),
--      noOutputsDisabled   (5),
--      noInputsDisabled    (6),
--      forced              (7),
--      appPresent          (8),
--      ioFault             (9),
--      puFault             (10),
--      powFault            (11),
--      memFault            (12),
--      comFault            (13),
--      implementerFault    (14)
    Attribute: Access Method = non-public
```

The semantic of the bits of the variable P_PCSTATE is derived from the health and state information of the PC subsystems, see 5.1.4:

- good – If TRUE, this attribute indicates the health of the PC system and of all its subsystems. This attribute is TRUE if and only if all subsystems of the PC indicate GOOD health condition;
- warning – If TRUE, this attribute indicates that at least one subsystem of the PC indicates a WARNING health condition and no subsystem indicates a "BAD" health condition;
- bad – If TRUE, this attribute indicates that at least one subsystem indicates a BAD health condition;
- running – If TRUE, this attribute indicates if at least one part of the user application has been loaded and is under control of the PC;
- localControl – If TRUE, this attribute indicates that local override control is active. If active, the ability to control a PC and its subsystems from the network may be limited. For example, this could be closely tied to the use of a local key switch;
- noOutputsDisabled – If TRUE, this attribute indicates that the PC can change the physical state of all outputs as a result of application program execution or other means. If not TRUE, the physical state of some of the outputs is not affected (logical state may be affected). This is typically used in the debugging of application programs in the PC;
- noInputsDisabled – If TRUE, this attribute indicates that the PC can access the physical state of all inputs as a result of application program execution or other means. If not TRUE, the physical state of some inputs cannot be accessed. This is typically used in the debugging of application programs where the inputs can be simulated;
- forced – If TRUE, this attribute indicates that at least one I/O point or variable associated with the PC has been forced. When a variable is forced, the application program shall use the value specified by the PADT instead of the value generated by the normal program execution;
- appPresent – If TRUE, this attribute indicates that at least one user application is present in the PC;