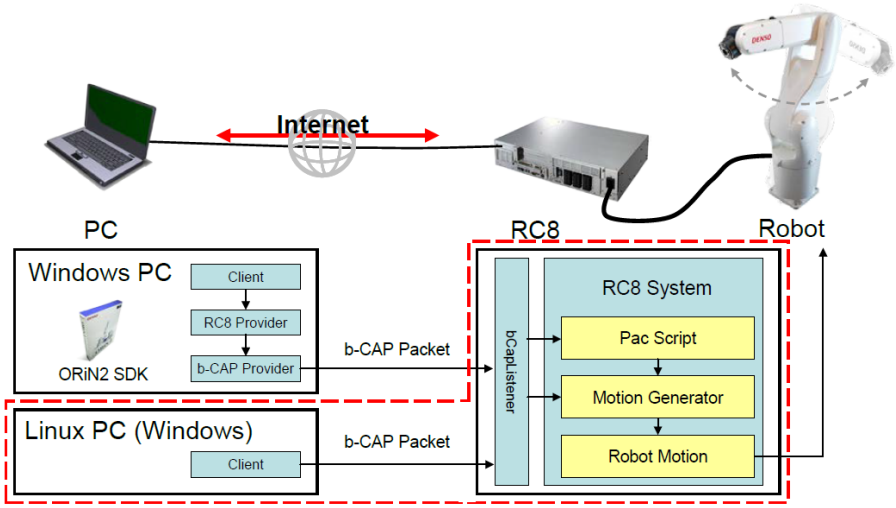
RC8 Provider

For DENSO Robot RC8

Version 1.1.5

User's Guide

(b-CAP Client Command Reference)



Version	Date	Content
1.1.5	2016-03-31	First Edition

Contents

1.1 System requirements and versions assumed in this document 10
1.2 Information sources for your reference 10
2. Environment Setup for Application Development 10
2.1 Setup of PC development environment 10
2.1.1 Automatic installation of RC8 provider 10
2.1.2 Manual installation of RC8 provider 10
2.2 Setup of RC8 controller 10
2.2.1 Emergency stop device position 10
2.2.2 Preparation of hardware 10
2.2.3 Setup of system parameters 10
2.2.3.1 Setup using a teach pendant 10
2.2.3.2 Setup using a mini teach pendant 10
2.3 Operation check using CaoTester 10
2.3.1 Check of variable access 10
2.3.2 Check that the motor is ON 10
3. Basic Knowledge on RC8 programing 10
3.2 Outline of RC8 provider 10
3.1.1 Functions provided by RC8 provider 10
3.1.2 System configuration of RC8 provider 10
3.1.2.1 Configuration of Cao engine and Cao provider 10
3.1.3 HRESULT and handling of errors 10
3.1.4 Handling of property definitions 10
3.1.5 Execute method and runtime binding 10
4. RC8 Programming Using the Provider 11
4.1 RC8 controller variable access 11
4.1.1 Connection 11
4.1.2 Variable read/write access 11
4.1.3 Disconnection 11
4.1.4 Sample program 11
4.2 Task control with RC8 controller 11
4.2.1 Connection 11
4.2.2 Start/Stop of a task 11
4.2.3 Sample program 11

4.3 Robot control with RC8 controller 11
4.3.1 Connection 11
4.3.2 Getting and release of arm control authority 11
4.3.3 Start and stop of the motor 11
4.3.4 Move and stop of the robot 11
4.3.5 Sample program 11
5. Command Reference 12
5.1 List of commands 12
5.2 Methods and properties 12
5.2.1 CaoWorkspace::AddController method 12
5.2.2.1 When you establish multiple connections with RC8 controller 12
5.2.2 CaoController::AddFile method 12
5.2.3 CaoController::AddRobot method 13
5.2.4 CaoController::AddTask method 13
5.2.5 CaoController::AddVariable method 14
5.2.6 CaoController::AddExtension method 16
5.2.7 CaoController::get_Name property 16
5.2.8 CaoController:: get_FileNames property 16
5.2.9 CaoController:: get_TaskName property 17
5.2.10 CaoController:: get_VariableNames property 18
5.2.11 CaoController:: Execute method 18
5.2.11.1 CaoController::Execute(“ClearError”) command 19
5.2.11.2 CaoController::Execute(“GetErrorDescription”) command 19
5.2.11.3 CaoController::Execute(“KillAll”) command 19
5.2.11.4 CaoController::Execute(“KillAllTsr”) command 20
5.2.11.5 CaoController::Execute(“RunAllTsr”) command 20
5.2.11.6 CaoController::Execute(“SuspendAll”) command 21
5.2.11.7 CaoController::Execute(“StepStopAll”) command 21
5.2.11.8 CaoController::Execute(“ContinueStartAll”) command 21
5.2.11.9 CaoController::Execute(“GetErrorLogCount”) command 22
5.2.11.10 CaoController::Execute(“GetErrorLog”) command 22
5.2.11.11 CaoController::Execute(“GetOprLogCount”) command 23
5.2.11.12 CaoController::Execute(“GetOprLog”) command 24
5.2.11.13 CaoController::Execute(“GetPublicValue”) command 25

5.2.11.14 CaoController::Execute(“SetPublicValue”) command 27
5.2.11.15 CaoController::Execute(“SysState”) command 30
5.2.11.16 CaoController::Execute(“SysInfo”) command 30
5.2.11.17 CaoController::Execute(“SetAllDummyIO”) command 31
5.2.11.18 CaoController::Execute(“GetCurErrorCount”) command 31
5.2.11.19 CaoController::Execute(“GetCurErrorInfo”) command 32
5.2.12 CaoFile::AddFile method 33
5.2.13 CaoFile::AddVariable method 33
5.2.14. CaoFile::get_VariableNames property 34
5.2.15. CaoFile::get_FileNames property 34
5.2.16. CaoFile::get_Size property 35
5.2.17. CaoFile::get_Value property 35
5.2.18. CaoFile::put_Value property 36
5.2.19. CaoRobot::Accelerate method 36
5.2.20. CaoRobot::AddVariable method 36
5.2.21. CaoRobot::get_VariableNames property 37
5.2.22. CaoRobot::Halt method 37
5.2.23. CaoRobot::Change method 37
5.2.24. CaoRobot::Drive method 39
5.2.25. CaoRobot::Move method 39
5.2.26. CaoRobot::Rotate method 40
5.2.27. CaoRobot::Speed method 42
5.2.28. CaoRobot::Execute method 42
5.2.28.1. CaoRobot::Execute("TMul") command 43
5.2.28.2. CaoRobot::Execute("TInv") command 44
5.2.28.3. CaoRobot::Execute("TNorm") command 44
5.2.28.4. CaoRobot::Execute("J2T") command 45
5.2.28.5. CaoRobot::Execute("T2J") command 46
5.2.28.6. CaoRobot::Execute("J2P") command 46
5.2.28.7. CaoRobot::Execute("P2J") command 47
5.2.28.8. CaoRobot::Execute("T2P") command 48
5.2.28.9. CaoRobot::Execute("P2T") command 48
5.2.28.10. CaoRobot::Execute("Dev") command 49
5.2.28.11. CaoRobot::Execute("DevH") command 49

5.2.28.12. CaoRobot::Execute("OutRange") command 50
5.2.28.13. CaoRobot::Execute("MPS") command 50
5.2.28.14. CaoRobot::Execute("RPM") command 51
5.2.28.15. CaoRobot::Execute("DPS") command 52
5.2.28.16. CaoRobot::Execute("CurPos") command 53
5.2.28.17. CaoRobot::Execute("DestPos") command 53
5.2.28.18. CaoRobot::Execute ("CurPosEx") command 53
5.2.28.19. CaoRobot::Execute("DestPosEx") command 54
5.2.28.20. CaoRobot::Execute("HighCurPosEx") command 54
5.2.28.21. CaoRobot::Execute("CurJnt") command 55
5.2.28.22. CaoRobot::Execute("DestJnt") command 55
5.2.28.23. CaoRobot::Execute("CurJntEx") command 55
5.2.28.24. CaoRobot::Execute("DestJntEx") command 56
5.2.28.25. CaoRobot::Execute("HighCurJntEx") command 56
5.2.28.26. CaoRobot::Execute("CurTrn") command 56
5.2.28.27. CaoRobot::Execute("DestTrn") command 57
5.2.28.28. CaoRobot::Execute("CurTrnEx") command 57
5.2.28.29. CaoRobot::Execute("DestTrnEx") command 58
5.2.28.30. CaoRobot::Execute("HighCurTrnEx") command 58
5.2.28.31. CaoRobot::Execute("CurFig") command 58
5.2.28.32. CaoRobot::Execute("CurSpd") command 59
5.2.28.33. CaoRobot::Execute("CurAcc") command 59
5.2.28.34. CaoRobot::Execute("CurDec") command 59
5.2.28.35. CaoRobot::Execute("CurJSpd") command 60
5.2.28.36. CaoRobot::Execute("CurJAcc") command 60
5.2.28.37. CaoRobot::Execute("CurJDec") command 61
5.2.28.38. CaoRobot::Execute("StartLog") command 61
5.2.28.39. CaoRobot::Execute("StopLog") command 61
5.2.28.40. CaoRobot::Execute("ClearLog") command 62
5.2.28.41. CaoRobot::Execute("Motor") command 62
5.2.28.42. CaoRobot::Execute("ExtSpeed") command 63
5.2.28.43. CaoRobot::Execute("TakeArm") command 63
5.2.28.44. CaoRobot::Execute("GiveArm") command 64

5.2.28.45. CaoRobot::Execute("Draw") command 65
5.2.28.46. CaoRobot::Execute("Approach") command 65
5.2.28.47. CaoRobot::Execute("Depart") command 66
5.2.28.48. CaoRobot::Execute("DriveEx") command 67
5.2.28.49. CaoRobot::Execute("DriveAEx") command 68
5.2.28.50. CaoRobot::Execute("RotateH") command 68
5.2.28.51. CaoRobot::Execute("Arrive") command 69
5.2.28.52. CaoRobot::Execute("MotionSkip") command 69
5.2.28.53. CaoRobot::Execute("MotionComplete") command 70
5.2.28.54. CaoRobot::Execute("CurTool") command 71
5.2.28.55. CaoRobot::Execute("GetToolDef") command 71
5.2.28.56. CaoRobot::Execute("SetToolDef") command 72
5.2.28.57. CaoRobot::Execute("CurWork") command 73
5.2.28.58. CaoRobot::Execute("GetWorkDef") command 73
5.2.28.59. CaoRobot::Execute("SetWorkDef") command 74
5.2.28.60. CaoRobot::Execute("WorkAttribute") command 75
5.2.28.61. CaoRobot::Execute("GetAreaDef") command 75
5.2.28.62. CaoRobot::Execute("SetAreaDef") command 76
5.2.28.63. CaoRobot::Execute("SetArea") command 77
5.2.28.64. CaoRobot::Execute("ResetArea") command 77
5.2.28.65. CaoRobot::Execute("AreaSize") command 78
5.2.28.66. CaoRobot::Execute("GetAreaEnabled") command 79
5.2.28.67. CaoRobot::Execute("SetAreaEnabled") command 79
5.2.28.68. CaoRobot::Execute("AddPathPoint") command 80
5.2.28.69. CaoRobot::Execute("ClrPathPoint") command 80
5.2.28.70. CaoRobot::Execute("GetPathPoint") command 81
5.2.28.71. CaoRobot::Execute("LoadPathPoint") command 81
5.2.28.72. CaoRobot::Execute("GetPathPointCount ") command 82
5.2.28.73. CaoRobot::Execute("GetRobotTypeName") command 82
5.2.28.74. CaoRobot::Execute("ArchMove") command 82
5.2.28.75. CaoRobot::Execute("CrtMotionAllow") command 83
5.2.28.76. CaoRobot::Execute("EncMotionAllow") command 84
5.2.28.77. CaoRobot::Execute("EncMotionAllowJnt") command 85

5.2.28.78. CaoRobot::Execute("ErAlw") command 86
5.2.28.79. CaoRobot::Execute("ForceCtrl") command 87
5.2.28.80. CaoRobot::Execute("ForceParam") command 88
5.2.28.81. CaoRobot::Execute("ForceValue") command 89
5.2.28.82. CaoRobot::Execute("ForceWaitCondition") command 89
5.2.28.83. CaoRobot::Execute("ForceSensor") command 90
5.2.28.84. CaoRobot::Execute("ForceChangeTable") command 90
5.2.28.85. CaoRobot::Execute("GetSrvData") command 91
5.2.28.86. CaoRobot::Execute("GetSrvJntData") command 91
5.2.28.87. CaoRobot::Execute("GrvCtrl") command 92
5.2.28.88. CaoRobot::Execute("CurLmt") command 92
5.2.28.89. CaoRobot::Execute("Zforce") command 94
5.2.28.90. CaoRobot::Execute("GrvOffset") command 95
5.2.28.91. CaoRobot::Execute("HighPathAccuracy") command 95
5.2.28.92. CaoRobot::Execute("MotionTimeout") command 96
5.2.28.93. CaoRobot::Execute("SingularAvoid") command 97
5.2.28.94. CaoRobot::Execute("SpeedMode") command 98
5.2.28.95. CaoRobot::Execute("PayLoad") command 98
5.2.28.96. CaoRobot::Execute("GenerateNonStopPath ") command 99
5.2.28.97. CaoRobot::Execute("RobInfo") command 99
5.2.28.98. CaoRobot::Execute("SyncTimeStart") command 99
5.2.28.99. CaoRobot::Execute("SyncTimeEnd") command 101
5.2.28.100. CaoRobot::Execute("SyncMoveStart") command 102
5.2.28.101. CaoRobot::Execute("SyncMoveEnd") command 103
5.2.28.102. CaoRobot::Execute("SetBaseDef") command 105
5.2.28.103. CaoRobot::Execute("GetBaseDef") command 106
5.2.28.104. CaoRobot::Execute("SetHandIO") command 106
5.2.28.105. CaoRobot::Execute("GetHandIO") command 107
5.2.28.106. CaoRobot::Execute("StartServoLog") command 108
5.2.28.107. CaoRobot::Execute("ClearServoLog") command 108
5.2.28.108. CaoRobot::Execute("StopServoLog") command 108
5.2.28.109. CaoRobot::Execute("GetCtrlLogMaxTime") command 109
5.2.28.110. CaoRobot::Execute("SetCtrlLogMaxTime") command 109

5.2.28.111. CaoRobot::Execute("GetCtrlLogInterval") command 110
5.2.28.112. CaoRobot::Execute("SetCtrlLogInterval ") command 110
5.2.28.113. CaoRobot::Execute("DetectOn ") command 110
5.2.28.114. CaoRobot::Execute("DetectOff ") command 111
5.2.28.115. CaoRobot::Execute("GetPluralServoData ") command 112
5.2.28.116. CaoRobot::Execute("AngularTrigger ") command 112
5.2.29. CaoTask::AddVariable method 113
5.2.30. CaoTask::get_VariableNames property 113
5.2.31. CaoTask::Start method 113
5.2.32. CaoTask::Stop method 113
5.2.33. CaoTask::Execute method 113
5.2.33.1. CaoTask::Execute("GetStatus") command 113
5.2.33.2. CaoTask::Execute("GetThreadPriority") command 114
5.2.33.3. CaoTask::Execute("SetThreadPriority") command 114
5.2.34. CaoVariable::get_Value property 114
5.2.35. CaoVariable::put_Value property 114
5.2.36. CaoExtension::Execute method 114
5.2.36.1. Hand object - CaoExtension::Execute("Chuck") command 114
5.2.36.2. Hand object - CaoExtension::Execute("UnChuck") command 115
5.2.36.3. Hand object - CaoExtension::Execute("Motor") command 115
5.2.36.4. Hand object - CaoExtension::Execute("Org") command 116
5.2.36.5. Hand object - CaoExtension::Execute("MoveP") command 116
5.2.36.6. Hand object - CaoExtension::Execute("MoveA") command 116
5.2.36.7. Hand object - CaoExtension::Execute("MoveR") command 117
5.2.36.8. Hand object - CaoExtension::Execute("MoveAH") command 118
5.2.36.9. Hand object - CaoExtension::Execute("MoveRH") command 118
5.2.36.10. Hand object - CaoExtension::Execute("MoveH") command 119
5.2.36.11 Hand object – CaoExtension::Execute(“MoveZH”) command 119
5.2.36.12 Hand object – CaoExtension::Execute(“Stop”) command 120
5.2.36.13 Hand object – CaoExtension::Execute(“CurPos”) command 120
5.2.36.14 Hand object – CaoExtension::Execute(“GetPoint”) command 121
5.2.36.15 Hand object – CaoExtension::Execute(“get_EmgState”) command 121
5.2.36.16 Hand object – CaoExtension::Execute(“get_ZonState”) command 122

5.2.36.17 Hand object – CaoExtension::Execute("get_OrgState") command 122
5.2.36.18 Hand object – CaoExtension::Execute("get_HoldState") command 122
5.2.36.19 Hand object – CaoExtension::Execute("get_InposState") command 123
5.2.36.20 Hand object – CaoExtension::Execute("get_Error") command 123
5.2.36.21 Hand object – CaoExtension::Execute("get_BusyState") command 124
5.2.36.22 Hand object – CaoExtension::Execute("get_MotorState") command 124

1. Introduction

1.1 System requirements and versions assumed in this document

1.2 Information sources for your reference

2. Environment Setup for Application Development

2.1 Setup of PC development environment

2.1.1 Automatic installation of RC8 provider

2.1.2 Manual installation of RC8 provider

2.2 Setup of RC8 controller

2.2.1 Emergency stop device position

2.2.2 Preparation of hardware

2.2.3 Setup of system parameters

2.2.3.1 Setup using a teach pendant

2.2.3.2 Setup using a mini teach pendant

2.3 Operation check using CaoTester

2.3.1 Check of variable access

2.3.2 Check that the motor is ON

3. Basic Knowledge on RC8 programing

3.2 Outline of RC8 provider

3.1.1 Functions provided by RC8 provider

3.1.2 System configuration of RC8 provider

3.1.2.1 Configuration of Cao engine and Cao provider

3.1.3 HRESULT and handling of errors

3.1.4 Handling of property definitions

3.1.5 Execute method and runtime binding

4. RC8 Programming Using the Provider

4.1 RC8 controller variable access

4.1.1 Connection

4.1.2 Variable read/write access

4.1.3 Disconnection

4.1.4 Sample program

4.2 Task control with RC8 controller

4.2.1 Connection

4.2.2 Start/Stop of a task

4.2.3 Sample program

4.3 Robot control with RC8 controller

4.3.1 Connection

4.3.2 Getting and release of arm control authority

4.3.3 Start and stop of the motor

4.3.4 Move and stop of the robot

4.3.5 Sample program

5. Command Reference

5.1 List of commands

5.2 Methods and properties

5.2.1 CaoWorkspace::AddController method

Example – Create CaoController

```
int fd;                             //Socket Pointer
HRESULT hr;                         //Command Status
uint32_t hCtrl;                     //Controller Handler

//Open Client Socket
hr = bCap_Open_Client("tcp:" TARGET_RC8_IP, 1000, 3, &fd);
if (SUCCEEDED(hr))
{
 /* Send SERVICE_START Packet */
 bCap_ServiceStart(fd, NULL);
 //Get Controller Handler
 BSTR ctrl_name, ctrl_prov, ctrl_mach, ctrl_opt;       //Arguments:
 ctrl_name = SysAllocString(L"");                      //Name
 ctrl_prov = SysAllocString(L"CaoProv.DENSO.VRC");     //Provider
 ctrl_mach = SysAllocString(L"localhost");             //Machine
 ctrl_opt = SysAllocString(L"");                       //Option

 /* Connect to RC8 */
 hr = bCap_ControllerConnect(fd, ctrl_name, ctrl_prov, ctrl_mach, ctrl_opt, &hCtrl);

 if (SUCCEEDED(hr))
  printf("bCap_ControllerConnect Succeeded...\n");
 else
  printf("bCap_ControllerConnect Failed...\n");

 //Release Variables
 SysFreeString(ctrl_name);
 SysFreeString(ctrl_prov);
 SysFreeString(ctrl_mach);
 SysFreeString(ctrl_opt);
}
else
 printf("bCap_Open_Client Failed...\n");
```

5.2.2.1 When you establish multiple connections with RC8 controller

5.2.2 CaoController::AddFile method

Example – Get the content of a Pro1.pcs file

```
//Get File Handler
uint32_t hFile;                         //File Handler
BSTR file_name, file_opt;               //Arguments:
file_name = SysAllocString(L"Pro1.pcs"); //Name
file_opt = SysAllocString(L"");         //Option

/* Obtain File Reference */
hr = bCap_ControllerGetFile(fd, hCtrl, file_name, file_opt, &hFile);
```

```
if (SUCCEEDED(hr))
 printf("bCap_ControllerGetFile Succeeded...\n");
else
 printf("bCap_ControllerGetFile Failed...\n");

//Release Variables
SysFreeString(file_name);
SysFreeString(file_opt);
```

5.2.3 CaoController::AddRobot method

Example

```
uint32_t hRobot;                        //Robot Handler
//Get Robot Handler
BSTR rob_name, rob_opt;                 //Arguments:
rob_name = SysAllocString(L"Robot0");   //Name
rob_opt = SysAllocString(L"ID=0");      //Option

/* Obtain Robot Reference */
hr = bCap_ControllerGetRobot(fd, hCtrl, rob_name, rob_opt, &hRobot);
if (SUCCEEDED(hr))
 printf("bCap_ControllerGetRobot Succeeded...\n");
else
 printf("bCap_ControllerGetRobot Failed...\n");

//Release Variables
SysFreeString(rob_name);
SysFreeString(rob_opt);
```

5.2.4 CaoController::AddTask method

Example

```
//Get Task Handler
uint32_t hTask;                         //Task Handler
BSTR task_name, task_opt;               //Arguments:
task_name = SysAllocString(L"Pro1");    //Name
task_opt = SysAllocString(L"");         //Option

/* Obtain Task Reference */
hr = bCap_ControllerGetTask(fd, hCtrl, task_name, task_opt, &hTask);
if (SUCCEEDED(hr))
 printf("bCap_ControllerGetTask Succeeded...\n");
else
 printf("bCap_ControllerGetTask Failed...\n");

//Release Variables
SysFreeString(task_name);
SysFreeString(task_opt);
```

5.2.5 CaoController::AddVariable method

Example – Access to the 128th I/O variable

```
//Get Variable Handler
uint32_t hVariable;                     //Variable Handler
BSTR var_name, var_opt;                 //Arguments:
var_name = SysAllocString(L"IO128");    //Name
var_opt = SysAllocString(L"");          //Option

/* Obtain Variable Reference */
hr = bCap_ControllerGetVariable(fd, hCtrl, var_name, var_opt, &hVariable);
if (SUCCEEDED(hr))
{
 printf("bCap_ControllerGetVariable Succeeded...\n");

 //Write to Controller Variable
 VARIANT write_value;
 VariantInit(&write_value);
 write_value.boolVal = 1;
 write_value.vt = VT_BOOL;

 /* Write Value to Controller */
 hr = bCap_VariablePutValue(fd, hVariable, write_value);
 if (SUCCEEDED(hr))
 {
  printf("bCap_VariablePutValue Succeeded...\n");
  VARIANT read_value;
  VariantInit(&read_value);

  /* Read Value from Controller */
  hr = bCap_VariableGetValue(fd, hVariable, &read_value);
  if (SUCCEEDED(hr))
  {
   printf("bCapVariableGetValue Succeeded...\n");
   printf("%S value is %d \n", var_name, read_value.boolVal);
  }
  else
   printf("bCapVariableGetValue Failed...\n");

  //Release Variables
  VariantClear(&read_value);
 }
 else
  printf("bCapVariablePutValue Failed...\n");

 //Release Variable Handler
 bCap_VariableRelease(fd, &hVariable);

 //Release Variables
 VariantClear(&write_value);

}
else
 printf("bCap_ControllerGetVariable Failed...\n");

//Release Variables
SysFreeString(var_name);
SysFreeString(var_opt);
```

```
//Example #2-----------------------------------
var_name = SysAllocString(L"IO*"); //Name
var_opt = SysAllocString(L"");     //Option

/* Obtain Variable Reference */
hr = bCap_ControllerGetVariable(fd, hCtrl, var_name, var_opt, &hVariable);

if (SUCCEEDED(hr))
{
 printf("bCap_ControllerVariable Succeeded...\n");

 //Specify Variable Index ID
 VARIANT var_id;
 VariantInit(&var_id);
 var_id.intVal = 129;
 var_id.vt = VT_I4;

 /* Set Variable Inde ID */
 hr = bCap_VariablePutID(fd, hVariable, var_id);
 if (SUCCEEDED(hr))
 {
  printf("bCap_VariablePutID Succeeded...\n");

  //Write to Controller Variable
  VARIANT write_value;
  VariantInit(&write_value);
  write_value.boolVal = 1;
  write_value.vt = VT_BOOL;

  /* Write Value to Controller */
  hr = bCap_VariablePutValue(fd, hVariable, write_value);

  if (SUCCEEDED(hr))
  {
   printf("bCap_VariablePutValue Succeeded...\n");
   VARIANT read_value;
   VariantInit(&read_value);

   /* Read Value from Controller */
   hr = bCap_VariableGetValue(fd, hVariable, &read_value);
   if (SUCCEEDED(hr))
   {
    printf("bCapVariableGetValue Succeeded...\n");
    printf("%S value is %d \n", var_name, read_value.boolVal);
   }
   else
    printf("bCapVariableGetValue Failed...\n");

   //Release Variables
   VariantClear(&read_value);
  }
  else
   printf("bCapVariablePutValue Failed...\n");

  //Release Variables
  VariantClear(&write_value);
```

```
 }
 else
  printf("bCap_VariablePutID Failed...\n");
}
else
 printf("bCap_ControllerVariable Failed...\n");

//Release Variables
SysFreeString(var_name);
SysFreeString(var_opt);
VariantClear(&var_id);
```

5.2.6 CaoController::AddExtension method

Example

```
//Get Extension Handler
uint32_t hExtension;                    //Extension Handler
BSTR ext_name, ext_opt;                 //Arguments:
ext_name = SysAllocString(L"Hand0");    //Name
ext_opt = SysAllocString(L"");          //Option

/* Obtain Extension Reference */
hr = bCap_ControllerGetExtension(fd, hCtrl, ext_name, ext_opt, &hExtension);

if (SUCCEEDED(hr))
 printf("bCap_ControllerGetExtension Succeeded...\n");
else
 printf("bCap_ControllerGetExtension Failed...\n");

//Release Variables
SysFreeString(ext_name);
SysFreeString(ext_opt);
```

5.2.7 CaoController::get_Name property

Example – Display the automatically assigned controller name

```
//Get Controller Name
BSTR ctrl_name;
hr = bCap_ControllerGetName(fd, hCtrl, &ctrl_name);

if (SUCCEEDED(hr))
{
 printf("bCap_ControllerGetName Succeeded...\n");
 printf("Controller Name: %S \n", ctrl_name);
}
else
 printf("bCap_ControllerGetName Failed...\n");

//Release Variables
SysFreeString(ctrl_name);
```

5.2.8 CaoController:: get_FileNames property

Example – List the following file names in the root folder

```
//Get File Names
BSTR file_opt;                          //Arguments:
file_opt = SysAllocString(L"");         //Option
```

```
VARIANT file_names;
VariantInit(&file_names);

/* Get List of Files on RC8 */
hr = bCap_ControllerGetFileNames(fd, hCtrl, file_opt, &file_names);
if (SUCCEEDED(hr))
{
 printf("bCap_ControllerGetFileNames Succeeded...\n");

 //Print File List
 SAFEARRAY* file_list = NULL;
 file_list = V_ARRAY(&file_names);

 BSTR* file_name = NULL;
 hr = SafeArrayAccessData(file_list, (void**)&file_name);
 if (SUCCEEDED(hr))
 {
  long ubound, lbound;
  hr = SafeArrayGetUBound(file_list, 1, &ubound);
  hr = SafeArrayGetLBound(file_list, 1, &lbound);

  long cnt_elements = ubound - lbound + 1;

  printf("RC8 File List: \n");
  for (int i = 0; i < cnt_elements; i++)
  {
   printf("\t - %d: %S \n", i, file_name[i]);
  }
  SafeArrayUnaccessData(file_list);
 }

 //Release Variables
 SafeArrayDestroy(file_list);
}
else
 printf("bCap_ControllerGetFileNames Failed...\n");

//Release Variables
SysFreeString(file_opt);
VariantClear(&file_names);
```

5.2.9 CaoController:: get_TaskName property

Example – List task names

```
//Get Task Names
BSTR task_opt;                          //Arguments:
VARIANT task_names;
task_opt = SysAllocString(L"");         //Option
VariantInit(&task_names);               //Task List

/* Get List of Tasks on RC8 */
hr = bCap_ControllerGetTaskNames(fd, hCtrl, task_opt, &task_names);
if (SUCCEEDED(hr))
{
 printf("bCap_ControllerGetTaskNames Succeeded...\n");

 //Print Task List
```

```
 SAFEARRAY* task_list = NULL;
 task_list = V_ARRAY(&task_names);

 BSTR* task_name = NULL;
 hr = SafeArrayAccessData(task_list, (void**)&task_name);

 if (SUCCEEDED(hr))
 {
  long ubound, lbound;
  hr = SafeArrayGetUBound(task_list, 1, &ubound);
  hr = SafeArrayGetLBound(task_list, 1, &lbound);

  long cnt_elements = ubound - lbound + 1;
  printf("RC8 Task List: \n");
  for (int i = 0; i < cnt_elements; i++)
  {
   printf("\t - %d: %S \n", i, task_name[i]);
  }
  SafeArrayUnaccessData(task_list);
 }
}
else
 printf("bCap_ControllerGetTaskNames Failed...\n");

//Release Variables
SysFreeString(task_opt);
VariantClear(&task_names);
```

5.2.10 CaoController:: get_VariableNames property

5.2.11 CaoController:: Execute method

Example

```
//ExecuteTask
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"ClearError"); //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

/* Clear Error on RC8 */
hr = bCap_ControllerExecute(fd, hCtrl, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_ControllerExecute Succeeded... \n");
else
 printf("bCap_ControllerExecute Failed... \n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.11.1 CaoController::Execute("ClearError") command

Example

```
//Execute
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"ClearError"); //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

/* Clear Error on RC8 */
hr = bCap_ControllerExecute(fd, hCtrl, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_ControllerExecute Succeeded... \n");
else
 printf("bCap_ControllerExecute Failed... \n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.11.2 CaoController::Execute("GetErrorDescription") command

Example

```
BSTR exe_comm;                                      //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"GetErrorDescription");  //Command Name
VariantInit(&exe_param);                            //Command Parameters
VariantInit(&exe_result);                           //Command Result

//Error Description Parameters
exe_param.intVal = 0x83500003;    //Error Code
exe_param.vt = VT_I4;

/* Get Error Description */
hr = bCap_ControllerExecute(fd, hCtrl, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
{
 printf("bCap_ControllerExecute Succeeded...\n");
 printf("Error Description for Code 83500003: %S", exe_result.bstrVal);
}
else
 printf("bCap_ControllerExecute Failed... \n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.11.3 CaoController::Execute("KillAll") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"KillAll");  //Command Name
VariantInit(&exe_param);                //Command Parameters
```

```
VariantInit(&exe_result);                //Command Result

/* Kill All Programs */
hr = bCap_ControllerExecute(fd, hCtrl, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_ControllerExecute Succeeded...\n");
else
 printf("bCap_ControllerExecute Failed... \n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.11.4 CaoController::Execute("KillAllTsr") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"KillAllTsr");       //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

/* Kill All TSR Programs */
hr = bCap_ControllerExecute(fd, hCtrl, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_ControllerExecute Succeeded...\n");
else
 printf("bCap_ControllerExecute Failed... \n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.11.5 CaoController::Execute("RunAllTsr") command

Example

```
BSTR exe_comm;                           //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"RunAllTsr"); //Command Name
VariantInit(&exe_param);                 //Command Parameters
VariantInit(&exe_result);                //Command Result

/* Run All TSR Programs */
hr = bCap_ControllerExecute(fd, hCtrl, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_ControllerExecute Succeeded...\n");
else
 printf("bCap_ControllerExecute Failed... \n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.11.6 CaoController::Execute("SuspendAll") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"SuspendAll"); //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

/* Suspend All Programs */
hr = bCap_ControllerExecute(fd, hCtrl, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_ControllerExecute Succeeded...\n");
else
 printf("bCap_ControllerExecute Failed... \n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.11.7 CaoController::Execute("StepStopAll") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"StepStopAll");      //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

/* StepStop All Programs */
hr = bCap_ControllerExecute(fd, hCtrl, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_ControllerExecute Succeeded...\n");
else
 printf("bCap_ControllerExecute Failed... \n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.11.8 CaoController::Execute("ContinueStartAll") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"ContinueStartAll"); //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

/* Continue Start All Programs */
hr = bCap_ControllerExecute(fd, hCtrl, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_ControllerExecute Succeeded...\n");
else
 printf("bCap_ControllerExecute Failed... \n");
```

```
//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.11.9 CaoController::Execute("GetErrorLogCount") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"GetErrorLogCount"); //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

/* Get total number of erros in Error Log */
hr = bCap_ControllerExecute(fd, hCtrl, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
{
 printf("bCap_ControllerExecute Succeeded...\n");
 printf("Error Log Count: %ld \n", exe_result.intVal);
}
else
 printf("bCap_ControllerExecute Failed... \n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.11.10 CaoController::Execute("GetErrorLog") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"GetErrorLogCount"); //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

/* Get total number of erros in Error Log */
hr = bCap_ControllerExecute(fd, hCtrl, exe_comm, exe_param, &exe_result);

//Release Variable
SysFreeString(exe_comm);

if (SUCCEEDED(hr))
{
 printf("bCap_ControllerExecute Succeeded...\n");
 int error_ubound = exe_result.intVal;
 exe_comm = SysAllocString(L"GetErrorLog");     //Command Name

 printf("RC8 Error Log:_____ \n");
 printf("No. | Date | Call | Code | Message \n");

 for (int i = 0; i < error_ubound; i++)
 {
  exe_param.intVal = i;     //Error Log Index
```

```
  exe_param.vt = VT_I4;

  /* Get Error Information based on Index */
  hr = bCap_ControllerExecute(fd, hCtrl, exe_comm, exe_param, &exe_result);
  if (SUCCEEDED(hr))
  {
   //Print Error Info
   SAFEARRAY* log_list = NULL;
   log_list = V_ARRAY(&exe_result);

   VARIANT* error_log;
   hr = SafeArrayAccessData(log_list, (void**)&error_log);

   if (SUCCEEDED(hr))
   {
    printf("%d | ", i);
    printf("%d/%d/%d | ", error_log[1].intVal, error_log[2].intVal, error_log[4].intVal);
    printf("%d | ", error_log[14].intVal);
    printf("%X | ", error_log[13].intVal);
    printf("%S \n", error_log[12].bstrVal);

    SafeArrayUnaccessData(log_list);
   }
  }
  else
   printf("bCap_ControllerExecute Failed... \n");
 }
}
else
 printf("bCap_ControllerExecute Failed... \n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.11.11 CaoController::Execute("GetOprLogCount") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"GetOprLogCount");   //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

/* Get total number of operations in Operation Log */
hr = bCap_ControllerExecute(fd, hCtrl, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
{
 printf("bCap_ControllerExecute Succeeded...\n");
 printf("Operation Log Count: %d \n", exe_result.intVal);
}
else
 printf("bCap_ControllerExecute Failed... \n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
```

```
VariantClear(&exe_result);
```

5.2.11.12 CaoController::Execute("GetOprLog") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"GetOprLogCount");   //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

/* Get total number of operatoins in Operation Log */
hr = bCap_ControllerExecute(fd, hCtrl, exe_comm, exe_param, &exe_result);

//Release Variable
SysFreeString(exe_comm);

if (SUCCEEDED(hr))
{
 printf("bCap_ControllerExecute Succeeded...\n");
 int error_ubound = exe_result.intVal;
 exe_comm = SysAllocString(L"GetOprLog"); //Command Name

 printf("RC8 Operation Log:_____ \n");
 printf("No. | Date | Call | Code | Message \n");

 for (int i = 0; i < error_ubound; i++)
 {
  exe_param.intVal = i;     //Operation Log Index
  exe_param.vt = VT_I4;

  /* Get Error Information based on Index */
  hr = bCap_ControllerExecute(fd, hCtrl, exe_comm, exe_param, &exe_result);
  if (SUCCEEDED(hr))
  {
   //Print Operation Info
   SAFEARRAY* log_list = NULL;
   log_list = V_ARRAY(&exe_result);

   VARIANT* opr_log;
   hr = SafeArrayAccessData(log_list, (void**)&opr_log);

   if (SUCCEEDED(hr))
   {
    printf("%d | ", i);
    printf("%d/%d/%d | ", opr_log[1].intVal, opr_log[2].intVal, opr_log[4].intVal);
    printf("%d | ", opr_log[10].intVal);
    printf("%X | ", opr_log[0].intVal);
    printf("%S \n", opr_log[11].bstrVal);

    SafeArrayUnaccessData(log_list);
   }
  }
  else
   printf("bCap_ControllerExecute Failed... \n");
 }
}
```

```
else
 printf("bCap_ControllerExecute Failed... \n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.11.13 CaoController::Execute("GetPublicValue") command

Example – Reading Variable

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"GetPublicValue");   //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

//Populate parameter option
BSTR *param_data;
exe_param.vt = VT_BSTR | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_BSTR, 0, 2);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0] = SysAllocString(L"Pro1");            //Task Name
param_data[1] = SysAllocString(L"pblValue");        //Variable Name
SafeArrayUnaccessData(exe_param.parray);

/* Reading a Variable (Single Dimension) */
hr = bCap_ControllerExecute(fd, hCtrl, exe_comm, exe_param, &exe_result);

////Release Variable
for (int i = 0; i < 2; i++)
 SysFreeString(param_data[i]);

if (SUCCEEDED(hr))
{
 printf("hCap_ControllerExecute Succeeded...\n");
 printf("Variable Value: %d\n", exe_result.intVal);
}
else
 printf("hCap_ControllerExecute Failed...\n");
```

Example – Reading one element array

```
//Populate parameter option
VARIANT *param_data2;
exe_param.vt = VT_VARIANT | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_VARIANT, 0, 5);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data2);
param_data2[0].bstrVal = SysAllocString(L"Pro1");
param_data2[0].vt = VT_BSTR;
param_data2[1].bstrVal = SysAllocString(L"pblArrays");
param_data2[1].vt = VT_BSTR;
param_data2[2].intVal = 1;
param_data2[2].vt = VT_I4;
param_data2[3].intVal = 2;
param_data2[3].vt = VT_I4;
param_data2[4].intVal = 2;
param_data2[4].vt = VT_I4;
```

```
/* Reading a Variable (Element from an Array) */
hr = bCap_ControllerExecute(fd, hCtrl, exe_comm, exe_param, &exe_result);

//Release Variable
for (int i = 0; i < 2; i++)
 SysFreeString(param_data2[i].bstrVal);

if (SUCCEEDED(hr))
{
 printf("hCap_ControllerExecute Succeeded...\n");
 printf("Variable Value From Array: %d\n", exe_result.intVal);
}
else
 printf("hCap_ControllerExecute Failed...\n");
```

Example – Reading one dimensional array as a batch

```
//Populate parameter option
exe_param.vt = VT_BSTR | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_BSTR, 0, 2);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0] = SysAllocString(L"Pro1");                //Task Name
param_data[1] = SysAllocString(L"pblArray");            //Variable Name
SafeArrayUnaccessData(exe_param.parray);

/* Reading a Variable (Array) */
hr = bCap_ControllerExecute(fd, hCtrl, exe_comm, exe_param, &exe_result);

//Release Variable
for (int i = 0; i < 2; i++)
 SysFreeString(param_data[i]);

if (SUCCEEDED(hr))
{
 printf("bCap_ControllerExecute Succeeded...\n");

 //Print Operation Info
 SAFEARRAY* array_list = NULL;
 array_list = V_ARRAY(&exe_result);

 uint32_t* public_array;
 hr = SafeArrayAccessData(array_list, (void**)&public_array);
 if (SUCCEEDED(hr))
 {
  long ubound, lbound;
  hr = SafeArrayGetUBound(array_list, 1, &ubound);
  hr = SafeArrayGetLBound(array_list, 1, &lbound);

  long cnt_elements = ubound - lbound + 1;
  printf("ArrayValue: \n");

  for (int i = 0; i < cnt_elements; i++)
   printf("\t -%d: %d\n", i, public_array[i]);

  SafeArrayUnaccessData(array_list);
 }
}
else
 printf("bCap_ControllerExecute Failed... \n");
```

Example – Reading P-type variable

```
//Populate parameter option
exe_param.vt = VT_BSTR | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_BSTR, 0, 2);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0] = SysAllocString(L"Pro1");              //Task Name
param_data[1] = SysAllocString(L"pbPValue");          //Variable Name
SafeArrayUnaccessData(exe_param.parray);

/* Reading a Variable (Position) */
hr = bCap_ControllerExecute(fd, hCtrl, exe_comm, exe_param, &exe_result);

//Release Variable
for (int i = 0; i < 2; i++)
 SysFreeString(param_data[i]);

if (SUCCEEDED(hr))
{
 printf("bCap_ControllerExecute Succeeded...\n");

 //Print Operation Info
 SAFEARRAY* pos_list = NULL;
 pos_list = V_ARRAY(&exe_result);

 float* pos_array;
 hr = SafeArrayAccessData(pos_list, (void**)&pos_array);
 if (SUCCEEDED(hr))
 {
  long ubound, lbound;
  hr = SafeArrayGetUBound(pos_list, 1, &ubound);
  hr = SafeArrayGetLBound(pos_list, 1, &lbound);

  long cnt_elements = ubound - lbound + 1;
  printf("ArrayValue: \n");

  for (int i = 0; i < cnt_elements; i++)
   printf("\t -%d: %f\n", i, pos_array[i]);

  SafeArrayUnaccessData(pos_list);
 }
}
else
 printf("bCap_ControllerExecute Failed... \n");

//Release Variables
VariantClear(&exe_param);
VariantClear(&exe_result);
SysFreeString(exe_comm);
```

5.2.11.14 CaoController::Execute("SetPublicValue") command

Example – Writing Variable

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"SetPublicValue");   //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result
```

```
//Populate parameter option
VARIANT *param_data;
exe_param.vt = VT_VARIANT | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_VARIANT, 0, 3);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0].intVal = 1234;
param_data[0].vt = VT_I4;
param_data[1].bstrVal = SysAllocString(L"Pro1");
param_data[1].vt = VT_BSTR;
param_data[2].bstrVal = SysAllocString(L"pblValue");
param_data[2].vt = VT_BSTR;
SafeArrayUnaccessData(exe_param.parray);

/* Writing to a Variable (Single Dimension) */
hr = bCap_ControllerExecute(fd, hCtrl, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_ControllerExecute Succeeded...\n");
else
 printf("bCap_ControllerExecute Failed...\n");
```

Example – Writing one element of array

```
//Populate parameter option
exe_param.vt = VT_VARIANT | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_VARIANT, 0, 6);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0].intVal = 1234;
param_data[0].vt = VT_I4;
param_data[1].bstrVal = SysAllocString(L"Pro1");
param_data[1].vt = VT_BSTR;
param_data[2].bstrVal = SysAllocString(L"pblArrays");
param_data[2].vt = VT_BSTR;
param_data[3].intVal = 1;
param_data[3].vt = VT_I4;
param_data[4].intVal = 2;
param_data[4].vt = VT_I4;
param_data[5].intVal = 2;
param_data[5].vt = VT_I4;
SafeArrayUnaccessData(exe_param.parray);

/* Writing to a Variable (Element from an Array) */
hr = bCap_ControllerExecute(fd, hCtrl, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_ControllerExecute Succeeded...\n");
else
 printf("bCap_ControllerExecute Failed...\n");
```

Example – Writing one dimensional array as batch

```
//Populate parameter option
exe_param.vt = VT_VARIANT | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_VARIANT, 0, 3);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);

//Build PublicVariable Array
uint32_t *iArray;
param_data[0].vt = VT_I4 | VT_ARRAY;
param_data[0].parray = SafeArrayCreateVector(VT_I4, 0, 3);
hr = SafeArrayAccessData(param_data[0].parray, (void**)&iArray);
```

```
for (int i = 0; i < 3; i++)
 iArray[i] = i;

SafeArrayUnaccessData(param_data[0].parray);
param_data[1].bstrVal = SysAllocString(L"Pro1");
param_data[1].vt = VT_BSTR;
param_data[2].bstrVal = SysAllocString(L"pblArray");
param_data[2].vt = VT_BSTR;
SafeArrayUnaccessData(exe_param.parray);

/* Writing to a Variable (Array) */
hr = bCap_ControllerExecute(fd, hCtrl, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_ControllerExecute Succeeded...\n");
else
 printf("bCap_ControllerExecute Failed...\n");
```

Example – Writing P-type variable

```
//Populate parameter option
exe_param.vt = VT_VARIANT | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_VARIANT, 0, 3);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);

//Build PublicPositionVariable Array
float *fVals;
param_data[0].vt = VT_R4 | VT_ARRAY;
param_data[0].parray = SafeArrayCreateVector(VT_R4, 0, 7);
hr = SafeArrayAccessData(param_data[0].parray, (void**)&fVals);
fVals[0] = 1.0;
fVals[1] = 2.0;
fVals[2] = 3.0;
fVals[3] = 1.0;
fVals[4] = 2.0;
fVals[5] = 3.0;
fVals[6] = -1;
SafeArrayUnaccessData(param_data[0].parray);
param_data[1].bstrVal = SysAllocString(L"Pro1");
param_data[1].vt = VT_BSTR;
param_data[2].bstrVal = SysAllocString(L"pbPValue");
param_data[2].vt = VT_BSTR;
SafeArrayUnaccessData(exe_param.parray);

/* Writing to a Variable (Position Variable) */
hr = bCap_ControllerExecute(fd, hCtrl, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_ControllerExecute Succeeded...\n");
else
 printf("bCap_ControllerExecute Failed...\n");

//Release Variables
VariantClear(&exe_param);
VariantClear(&exe_result);
SysFreeString(exe_comm);
```

5.2.11.15 CaoController::Execute("SysState") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"SysState");  //Command Name
VariantInit(&exe_param);                 //Command Parameters
VariantInit(&exe_result);                //Command Result

/* Get RC8 Systate */
hr = bCap_ControllerExecute(fd, hCtrl, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
{
 printf("bCap_ControllerExecute Succeeded...\n");
 printf("Systate Return Value: %X \n", exe_result.intVal);
}
else
 printf("bCap_ControllerExecute Failed... \n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.11.16 CaoController::Execute("SysInfo") command

Example

```
BSTR exe_comm;                          //Arguments
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"SysInfo");   //Command Name
VariantInit(&exe_param);                 //Command Parameters
VariantInit(&exe_result);                //Command Result

/*Parameter Settings*/
exe_param.intVal = 0;              //Manufacturing Number
exe_param.vt = VT_I4;              //Integer Type

/* Get RC8 SysInfo */
hr = bCap_ControllerExecute(fd, hCtrl, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
{
 printf("bCap_ControllerExecute Succeeded...\n");
 printf("Manufacturing Number: %S \n",exe_result.bstrVal);
}
else
 printf("bCap_ControllerExecute Failed... \n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.11.17 CaoController::Execute("SetAllDummyIO") command

Example

```
BSTR exe_comm;                                  //Arguments
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"SetAllDummyIO");    //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

/*Parameter Settings*/
exe_param.intVal = 0;             //Reset Dummy IO
exe_param.vt = VT_I4;             //Integer Type

/* Send SetAllDummyIO Command */
hr = bCap_ControllerExecute(fd, hCtrl, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_ControllerExecute Succeeded...\n");
else
 printf("bCap_ControllerExecute Failed... \n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.11.18 CaoController::Execute("GetCurErrorCount") command

Example

```
BSTR exe_comm;                                  //Arguments
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"GetCurErrorInfo");  //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

/*Set Parameters */
exe_param.intVal = 0;
exe_param.vt = VT_I4;

/* Send GGetCurErrorInfo Command */
hr = bCap_ControllerExecute(fd, hCtrl, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
{
 printf("bCap_ControllerExecute Succeeded...\n");

 //Print Operation Info
 SAFEARRAY* error_list = NULL;
 error_list = V_ARRAY(&exe_result);

 VARIANT* error_info;
 hr = SafeArrayAccessData(error_list, (void**)&error_info);
 if (SUCCEEDED(hr))
 {
  printf("Current Error Information: \n");
  printf("\t - Error Code: %X \n", error_info[0].intVal);
  printf("\t - Error Message: %S \n", error_info[1].bstrVal);
  printf("\t - Sub Code: %X \n", error_info[2].intVal);
  printf("\t - FileID&LineNo: %X \n", error_info[3].intVal);
```

```
  printf("\t - Program name: %S \n", error_info[4].bstrVal);
  printf("\t - Line number: %d \n", error_info[5].intVal);
  printf("\t - FileID: %d \n", error_info[6].intVal);

  SafeArrayUnaccessData(error_list);
 }
}
else
 printf("bCap_ControllerExecute Failed... \n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.11.19 CaoController::Execute("GetCurErrorInfo") command

Example

```
BSTR exe_comm;                                  //Arguments
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"GetCurErrorInfo");  //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

/*Set Parameters */
exe_param.intVal = 0;
exe_param.vt = VT_I4;

/* Send GGetCurErrorInfo Command */
hr = bCap_ControllerExecute(fd, hCtrl, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
{
 printf("bCap_ControllerExecute Succeeded...\n");

 //Print Operation Info
 SAFEARRAY* error_list = NULL;
 error_list = V_ARRAY(&exe_result);

 VARIANT* error_info;
 hr = SafeArrayAccessData(error_list, (void**)&error_info);
 if (SUCCEEDED(hr))
 {
  printf("Current Error Information: \n");
  printf("\t - Error Code: %X \n", error_info[0].intVal);
  printf("\t - Error Message: %S \n", error_info[1].bstrVal);
  printf("\t - Sub Code: %X \n", error_info[2].intVal);
  printf("\t - FileID&LineNo: %X \n", error_info[3].intVal);
  printf("\t - Program name: %S \n", error_info[4].bstrVal);
  printf("\t - Line number: %d \n", error_info[5].intVal);
  printf("\t - FileID: %d \n", error_info[6].intVal);

  SafeArrayUnaccessData(error_list);
 }
}
else
 printf("bCap_ControllerExecute Failed... \n");
```

```
//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.12 CaoFile::AddFile method

Example – Display the size of Pro1.pcs file in the User folder

```
//Get File Handler
BSTR file_name, file_opt;                                   //Arguments:
file_name = SysAllocString(L"User\\");          //Name
file_opt = SysAllocString(L"");                             //Option

/* Obtain File Reference */
hr = bCap_ControllerGetFile(fd, hCtrl, file_name, file_opt, &hFile);
if (SUCCEEDED(hr))
{
 printf("bCap_ControllerGetFile Succeeded...\n");

 BSTR file_name2;                                                   //Arguments:
 file_name2 = SysAllocString(L"Pro1.pcs");      //Name

 /* Add File */
 hr = bCap_FileGetFile(fd, hFile, file_name2, file_opt, &hFile2);
 if (SUCCEEDED(hr))
 {
  printf("bCap_FileGetFile Succeeded... \n");

  //Get File Size
  uint32_t *file_size;
  hr = bCap_FileGetSize(fd, hFile2, &file_size);
  if (SUCCEEDED(hr))
  {
   printf("bCap_FileGetSize Succeeded... \n");
   printf("Program Size: %d \n", file_size);
  }
  else
   printf("bCap_FileGetSize Failed... \n");
 }
 else
  printf("bCap_FileGetFile Failed... \n");

 //Release Variables
 SysFreeString(file_name2);
}
else
 printf("bCap_ControllerGetFile Failed...\n");

//Release Variables
SysFreeString(file_name);
SysFreeString(file_opt);
```

5.2.13 CaoFile::AddVariable method

Example – Get the CRC of the Pro1.pcs file

```
//Get File Handler
BSTR file_name, file_opt;                 //Arguments:
file_name = SysAllocString(L"Pro1.pcs"); //Name
file_opt = SysAllocString(L"");          //Option

/* Obtain File Reference */
hr = bCap_ControllerGetFile(fd, hCtrl, file_name, file_opt, &hFile);
if (SUCCEEDED(hr))
{
 printf("bCap_ControllerGetFile Succeeded... \n");
 BSTR var_name, var_opt;                  //Arguments:
 var_name = SysAllocString(L"@CRC");      //Name
 var_opt = SysAllocString(L"");           //Option

 /* Obtain File Variable Handler */
 hr = bCap_FileGetVariable(fd, hFile, var_name, var_opt, &hVariable);
 if (SUCCEEDED(hr))
 {
  printf("bCap_FileGetVariable Succeeded... \n");

  /* Read CRC Value */
  VARIANT read_value;
  hr = bCap_VariableGetValue(fd, hVariable, &read_value);
  if (SUCCEEDED(hr))
  {
   printf("bCap_VariableGetValue Succeeded... \n");
   printf("CRC Value: %X \n");
  }
  else
   printf("bCap_VariableGetValue Failed... \n");

  //Release Variables
  VariantClear(&read_value);
 }
 else
  printf("bCap_FileGetVariable Failed... \n");

 //Release Variables
 SysFreeString(var_name);
 SysFreeString(var_opt);
}
else
 printf("bCap_ControllerGetFile Failed... \n");

//Release Variables
SysFreeString(file_name);
SysFreeString(file_opt);
```

5.2.14. CaoFile::get_VariableNames property

5.2.15. CaoFile::get_FileNames property

5.2.16. CaoFile::get_Size property

Example – Get the size of Pro1.pcs file

```
//Get File Handler
BSTR file_name, file_opt;                              //Arguments:
file_name = SysAllocString(L"Pro1.pcs"); //Name
file_opt = SysAllocString(L"");                        //Option

/* Obtain File Reference */
hr = bCap_ControllerGetFile(fd, hCtrl, file_name, file_opt, &hFile);
if (SUCCEEDED(hr))
{
 //Get File Size
 uint32_t file_size;
 hr = bCap_FileGetSize(fd, hFile, &file_size);
 if (SUCCEEDED(hr))
 {
  printf("bCap_FileGetSize Succeeded... \n");
  printf("Program Size: %d \n", file_size);
 }
 else
  printf("bCap_FileGetSize Failed... \n");
}
else
 printf("bCap_ControllerGetFile Failed... \n");

//Release Variables
SysFreeString(file_name);
SysFreeString(file_opt);
```

5.2.17. CaoFile::get_Value property

Example – Get contents of Pro1.pcs file

```
//Get File Handler
BSTR file_name, file_opt;                 //Arguments:
file_name = SysAllocString(L"Pro1.pcs"); //Name
file_opt = SysAllocString(L"");           //Option

/* Obtain File Reference */
hr = bCap_ControllerGetFile(fd, hCtrl, file_name, file_opt, &hFile);
if (SUCCEEDED(hr))
{
 printf("bCap_ControllerGetFile Succeeded... \n");
 VARIANT file_value;

 /*Get File Contents */
 hr = bCap_FileGetValue(fd, hFile, &file_value);
 if (SUCCEEDED(hr))
 {
  printf("bCap_FileGetValue Succeeded... \n");
  printf("Pro1 Contents: \n%S", file_value.bstrVal);
 }
 else
  printf("bCap_FileGetValue Failed... \n");

 //Release Variables
 VariantClear(&file_value);
```

```
}
else
 printf("bCap_ControllerGetFile Succeeded... \n");

//Release Variables
SysFreeString(file_name);
SysFreeString(file_opt);
```

5.2.18. CaoFile::put_Value property

5.2.19. CaoRobot::Accelerate method

Example

```
//Acceleration = 50%, Deceleration = No Change
hr = bCap_RobotAccelerate(fd, hRobot, 0, 50, -1);
if (SUCCEEDED(hr))
 printf("bCap_RobotAccelerate Succeeded...\n");
else
 printf("bCap_RobotAccelerate Failed...\n");

//Acceleration = 50%, Deceleration = No Change
hr = bCap_RobotAccelerate(fd, hRobot, 0, -1, 60);
if (SUCCEEDED(hr))
 printf("bCap_RobotAccelerate Succeeded...\n");
else
 printf("bCap_RobotAccelerate Failed...\n");
```

5.2.20. CaoRobot::AddVariable method

Example – Refer to the current robot position (P type)

```
//Get Robot Handler
BSTR rob_name, rob_opt;                          //Arguments:
rob_name = SysAllocString(L"Arm0");       //Name
rob_opt = SysAllocString(L"");            //Option

/* Obtain Robot Reference */
hr = bCap_ControllerGetRobot(fd, hCtrl, rob_name, rob_opt, &hRobot);
if (SUCCEEDED(hr))
{
 printf("bCap_ControllerGetRobot Succeeded...\n");

 //Get Robot Variable Handler
 BSTR var_name, var_opt;           //Arguments:
 var_name = SysAllocString(L"@CURRENT_POSITION");
 var_opt = SysAllocString(L"");

 /* Obtain Robot Varaible Reference */
 hr = bCap_RobotGetVariable(fd, hRobot, var_name, var_opt, &hVariable);
 if (SUCCEEDED(hr))
 {
  printf("bCap_RobotGetVariable Succeeded...\n");
  /* Read Current Position Value */
  VARIANT read_value;
  hr = bCap_VariableGetValue(fd, hVariable, &read_value);
  if (SUCCEEDED(hr))
  {
   printf("bCap_VariableGetValue Succeeded... \n");
   //Print Operation Info
```

```
   SAFEARRAY* pos_list = NULL;
   pos_list = V_ARRAY(&read_value);

   double* pos_array;
   hr = SafeArrayAccessData(pos_list, (void**)&pos_array);
   if (SUCCEEDED(hr))
   {
    printf("Current Position Value:\n");
    printf("\t -X: %f\n",pos_array[0]);
    printf("\t -Y: %f\n",pos_array[1]);
    printf("\t -Z: %f\n",pos_array[2]);

    SafeArrayUnaccessData(pos_list);
   }
  }
  else
   printf("bCap_VariableGetValue Failed... \n");

  //Release Variables
  VariantClear(&read_value);
 }
 else
  printf("bCap_RobotGetVariable Failed...\n");

 //Release Variables
 SysFreeString(var_name);
 SysFreeString(var_opt);
}
else
 printf("bCap_ControllerGetRobot Failed...\n");

//Release Variables
SysFreeString(rob_name);
SysFreeString(rob_opt);
```

5.2.21. CaoRobot::get_VariableNames property

5.2.22. CaoRobot::Halt method

5.2.23. CaoRobot::Change method

Example

```
//Get Robot Handler
BSTR rob_name, rob_opt;                    //Arguments:
rob_name = SysAllocString(L"Arm0");        //Name
rob_opt = SysAllocString(L"");             //Option

/* Obtain Robot Reference */
hr = bCap_ControllerGetRobot(fd, hCtrl, rob_name, rob_opt, &hRobot);
if (SUCCEEDED(hr))
{
 printf("bCap_ControllerGetRobot Succeeded... \n");
 //ExecuteTask
 BSTR exe_comm;                            //Arguments:
 VARIANT exe_param, exe_result;
 exe_comm = SysAllocString(L"TakeArm");    //Command Name
 VariantInit(&exe_param);                  //Command Parameters
 VariantInit(&exe_result);                 //Command Result
```

```
//Populate parameter option
BSTR *param_data;
exe_param.vt = VT_I4 | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_BSTR, 0, 2);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0] = 0;         //Arm Group Number
param_data[1] = 0;         //Keep Value
SafeArrayUnaccessData(exe_param.parray);

/* Take Arm */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
{
 printf("bCap_RobotExecute Succeeded... \n");

 /* ChangeTool */
 hr = bCap_RobotChange(fd, hRobot, L"Tool1");
 if (SUCCEEDED(hr))
  printf("bCap_RobotChange Succeeded... \n");
 else
  printf("bCap_RobotChange Failed... \n");

 /* ChangeWork */
 hr = bCap_RobotChange(fd, hRobot, L"Work1");
 if (SUCCEEDED(hr))
  printf("bCap_RobotChange Succeeded... \n");
 else
  printf("bCap_RobotChange Failed... \n");

 /* Move */
 VARIANT move_pos;
 move_pos.bstrVal = SysAllocString(L"P10");
 move_pos.vt = VT_BSTR;
 hr = bCap_RobotMove(fd, hRobot, 1, move_pos, L"");
 if (SUCCEEDED(hr))
  printf("bCap_RobotMove Succeeded...\n");
 else
  printf("bCap_RobotMove Failed... \n");

 //Release Variables
 VariantClear(&move_pos);

 /* Givearm */
 exe_comm = SysAllocString(L"GiveArm");  //Command Name
 VariantInit(&exe_param);                        //Command Parameters
 VariantInit(&exe_result);                       //Command Result

 hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
 if (SUCCEEDED(hr))
  printf("bCap_RobotExecute Succeeded... \n");
 else
  printf("bCap_RobotExecute Failed... \n");
}
else
 printf("bCap_ControllerGetRobot Failed... \n");
```

```
 //Release Variables
 SysFreeString(exe_comm);
 VariantClear(&exe_param);
 VariantClear(&exe_result);
}
else
 printf("bCap_ControllerGetRobot Succeeded... \n");

//Release Variables
SysFreeString(rob_name);
SysFreeString(rob_opt);
```

5.2.24. CaoRobot::Drive method

5.2.25. CaoRobot::Move method

Example 1

```
BSTR move_opt;                                  //Arguments:
VARIANT move_pos;
uint32_t move_int;
move_int = 1;                                   //Move Interpolation
move_pos.bstrVal = SysAllocString(L"@P P1");    //Move Position
move_pos.vt = VT_BSTR;
move_opt = SysAllocString(L"NEXT");             //Move Option

/* Start Robot Motion */
hr = bCap_RobotMove(fd, hRobot, move_int, move_pos, move_opt);
if (SUCCEEDED(hr))
 printf("bCap_RobotMove Succeeded...\n");
else
 printf("bCap_RobotMove Failed... \n");
```

Example 2

```
move_int = 3;                                   //Move Interpolation
move_pos.bstrVal = SysAllocString(L"P1, @E P2");//Move Position
move_pos.vt = VT_BSTR;
move_opt = SysAllocString(L"");                 //Move Option

/* Start Robot Motion */
hr = bCap_RobotMove(fd, hRobot, move_int, move_pos, move_opt);
if (SUCCEEDED(hr))
 printf("bCap_RobotMove Succeeded...\n");
else
 printf("bCap_RobotMove Failed... \n");
```

Example 3

```
move_int = 2;                     //Move Interpolation

//Move Position
move_pos.bstrVal = SysAllocString(L"@0 P(307.1856, -157.8244, 107.0714, 160, 0, 0, 1)");
move_pos.vt = VT_BSTR;
move_opt = SysAllocString(L"");    //Move Option

/* Start Robot Motion */
hr = bCap_RobotMove(fd, hRobot, move_int, move_pos, move_opt);
if (SUCCEEDED(hr))
 printf("bCap_RobotMove Succeeded...\n");
```

```
else
 printf("bCap_RobotMove Failed... \n");
```

Example 4

```
move_int = 4;                                   //Move Interpolation
move_pos.bstrVal = SysAllocString(L"@E 2");     //Move Position
move_pos.vt = VT_BSTR;
move_opt = SysAllocString(L"");                 //Move Option

/* Start Robot Motion */
hr = bCap_RobotMove(fd, hRobot, move_int, move_pos, move_opt);
if (SUCCEEDED(hr))
 printf("bCap_RobotMove Succeeded...\n");
else
 printf("bCap_RobotMove Failed... \n");
```

Example 5

```
move_int = 1;                                                   //Move Interpolation
move_pos.bstrVal = SysAllocString(L"@P P10 EX((7,30.5))");      //Move Position
move_pos.vt = VT_BSTR;
move_opt = SysAllocString(L"NEXT");                             //Move Option

/* Start Robot Motion */
hr = bCap_RobotMove(fd, hRobot, move_int, move_pos, move_opt);
if (SUCCEEDED(hr))
 printf("bCap_RobotMove Succeeded...\n");
else
 printf("bCap_RobotMove Failed... \n");
```

Example 6

```
move_int = 3;                                                       //Move Interpolation
move_pos.bstrVal = SysAllocString(L"@E P20 EXA((7,30.8),(8,90.5))"); //Move Position
move_pos.vt = VT_BSTR;
move_opt = SysAllocString(L"");                                     //Move Option

/* Start Robot Motion */
hr = bCap_RobotMove(fd, hRobot, move_int, move_pos, move_opt);
if (SUCCEEDED(hr))
 printf("bCap_RobotMove Succeeded...\n");
else
 printf("bCap_RobotMove Failed... \n");

//Release Variables
SysFreeString(move_opt);
VariantClear(&move_opt);
```

5.2.26. CaoRobot::Rotate method

Example 1

```
VARIANT rot_surface, rot_pivot;                         //Arguments
float rot_angle;
BSTR rot_opt;
rot_surface.bstrVal = SysAllocString(L"V1, V2, V3");    //Rotation Surface
rot_surface.vt = VT_BSTR;
rot_angle = 45.8;                                       //Angle (deg)
rot_pivot.bstrVal = SysAllocString(L"V4");              //Rotation Center
rot_pivot.vt = VT_BSTR;
```

```
rot_opt = SysAllocString(L"@E");                        //Rotation Option (Pass)

/* Rotate */
hr = bCap_RobotRotate(fd, hRobot, rot_surface, rot_angle, rot_pivot, rot_opt);
if (SUCCEEDED(hr))
 printf("bCap_RobotRotate Succeeded...\n");
else
 printf("bCap_RobotRotate Failed... \n");
```

Example 2

```
rot_surface.bstrVal = SysAllocString(L"V(0,0,1), V(0,1,0), V(0,0,0)");//Rotation Surface
rot_surface.vt = VT_BSTR;
rot_angle = 30.0;                                                    //Angle (deg)
rot_pivot.bstrVal = SysAllocString(L"V(0,0,0)");                     //Rotation Center
rot_pivot.vt = VT_BSTR;
rot_opt = SysAllocString(L"@E, Pose=1, Next");                 //Rotation Option (Pass)

/* Rotate */
hr = bCap_RobotRotate(fd, hRobot, rot_surface, rot_angle, rot_pivot, rot_opt);
if (SUCCEEDED(hr))
 printf("bCap_RobotRotate Succeeded...\n");
else
 printf("bCap_RobotRotate Failed... \n");
```

Example 3

```
rot_surface.bstrVal = SysAllocString(L"XY");            //Rotation Surface
rot_surface.vt = VT_BSTR;
rot_angle = 90.0;                                       //Angle (deg)
rot_pivot.bstrVal = SysAllocString(L"V(0,0,0)");        //Rotation Center
rot_pivot.vt = VT_BSTR;
rot_opt = SysAllocString(L"@P");                        //Rotation Option (Pass)

/* Rotate */
hr = bCap_RobotRotate(fd, hRobot, rot_surface, rot_angle, rot_pivot, rot_opt);
if (SUCCEEDED(hr))
 printf("bCap_RobotRotate Succeeded...\n");
else
 printf("bCap_RobotRotate Failed... \n");
```

Example 4

```
rot_surface.bstrVal = SysAllocString(L"XYH");           //Rotation Surface
rot_surface.vt = VT_BSTR;
rot_angle = -45.0;                                      //Angle (deg)
rot_pivot.bstrVal = SysAllocString(L"V(250,0,0)");      //Rotation Center
rot_pivot.vt = VT_BSTR;
rot_opt = SysAllocString(L"@150");                      //Rotation Option (Pass)

/* Rotate */
hr = bCap_RobotRotate(fd, hRobot, rot_surface, rot_angle, rot_pivot, rot_opt);
if (SUCCEEDED(hr))
 printf("bCap_RobotRotate Succeeded...\n");
else
 printf("bCap_RobotRotate Failed... \n");

//Release Variables
VariantClear(&rot_surface);
VariantClear(&rot_pivot);
SysFreeString(rot_opt);
```

5.2.27. CaoRobot::Speed method

Example

```
//TakeArm Command has to be sent before Speed

int32_t speed_axis = -1;          //Axis Number
float speed_val = 85;             //Speed Value (%)
hr = bCap_RobotSpeed(fd, hRobot, speed_axis, speed_val);
if (SUCCEEDED(hr))
 printf("bCap_RobotSpeed Succeeded...\n");
else
 printf("bCap_RobotSpeed Failed...\n");
```

5.2.28. CaoRobot::Execute method

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"GetSrvJntData");    //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

//Populate parameter option
uint32_t *param_data;
exe_param.vt = VT_I4 | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_I4, 0, 2);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0] = 1;
param_data[1] = 6;
SafeArrayUnaccessData(exe_param.parray);

/* Execute GetJntData */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

exe_comm = SysAllocString(L"ExtSpeed");  //Command Name
VariantInit(&exe_param);                 //Command Parameters
VariantInit(&exe_result);                //Command Result

//Populate parameter option
float *param_data2;
exe_param.vt = VT_R4 | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_R4, 0, 3);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data2);
param_data2[0] = 50.0;
param_data2[1] = 25.0;
param_data2[2] = 25.0;
SafeArrayUnaccessData(exe_param.parray);

/* Execute ExtSpeed */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
```

```
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.1. CaoRobot::Execute("TMul") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"TMul");     //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

//Populate parameter option
BSTR *param_data;
exe_param.vt = VT_BSTR | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_BSTR, 0, 2);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0] = SysAllocString(L"T10");
param_data[1] = SysAllocString(L"T20");
SafeArrayUnaccessData(exe_param.parray);

/* Calculate by specifying the T type index */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Populate parameter option
exe_param.vt = VT_BSTR | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_BSTR, 0, 2);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0] = SysAllocString(L"T(400,500,400,1,0,0,0,1,0,5)");
param_data[1] = SysAllocString(L"T(100,0,0,1,0,0,0,1,0,-1)");
SafeArrayUnaccessData(exe_param.parray);

/* Calculate the T type element directly */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.2. CaoRobot::Execute("TInv") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"TInv");     //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

//Populate Parameter Option
exe_param.bstrVal = SysAllocString(L"T10");
exe_param.vt = VT_BSTR;

/* Inverse Matrix of T10 */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Populate Parameter Option
exe_param.bstrVal = SysAllocString(L"T(400,500,400,1,0,0,0,1,0,5)");
exe_param.vt = VT_BSTR;

/* Inverse Matrix of T10 */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.3. CaoRobot::Execute("TNorm") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"TNorm");    //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

//Populate Parameter Option
exe_param.bstrVal = SysAllocString(L"T10");
exe_param.vt = VT_BSTR;

/* Normalization of T10 */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");
```

```
//Populate Parameter Option
exe_param.bstrVal = SysAllocString(L"T(400,500,400,1,0,0,0,1,0,5)");
exe_param.vt = VT_BSTR;

/* Calculate by specifying the T type element directly */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.4. CaoRobot::Execute("J2T") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"J2T");      //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

//Populate Parameter Option
exe_param.bstrVal = SysAllocString(L"J10");
exe_param.vt = VT_BSTR;

/* Transform J10 value to T type data */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Populate Parameter Option
exe_param.bstrVal = SysAllocString(L"J(90,90,90,0,0,0)");
exe_param.vt = VT_BSTR;

/* Transform J10 value to T type data */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.5. CaoRobot::Execute("T2J") command

Example

```
BSTR exe_comm;                      //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"T2J"); //Command Name
VariantInit(&exe_param);            //Command Parameters
VariantInit(&exe_result);           //Command Result

//Populate Parameter Option
exe_param.bstrVal = SysAllocString(L"T10");
exe_param.vt = VT_BSTR;

/* Transform T10 value to J type data */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Populate Parameter Option
exe_param.bstrVal = SysAllocString(L"T(400,400,500,1,0,0,0,1,0,5)");
exe_param.vt = VT_BSTR;

/* Transform by specifying the T type element directly */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.6. CaoRobot::Execute("J2P") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"J2P");      //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

//Populate Parameter Option
exe_param.bstrVal = SysAllocString(L"J10");
exe_param.vt = VT_BSTR;

/* Transform J10 value to P type data */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");
```

```
//Populate Parameter Option
exe_param.bstrVal = SysAllocString(L"J(90,90,90,0,0,0)");
exe_param.vt = VT_BSTR;

/* Transform by specifying the J type data element directly */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.7. CaoRobot::Execute("P2J") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"P2J");      //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

//Populate Parameter Option
exe_param.bstrVal = SysAllocString(L"P10");
exe_param.vt = VT_BSTR;

/* Transform P10 value to J type data */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Populate Parameter Option
exe_param.bstrVal = SysAllocString(L"P(400,400,500,1,0,0,0,1,0,5)");
exe_param.vt = VT_BSTR;

/* Transform by specifying the P type element directly */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.8. CaoRobot::Execute("T2P") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"T2P");      //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

//Populate Parameter Option
exe_param.bstrVal = SysAllocString(L"T10");
exe_param.vt = VT_BSTR;

/* Transform T10 value to P type data */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Populate Parameter Option
exe_param.bstrVal = SysAllocString(L"T(400,400,500,1,0,0,0,1,0,5)");
exe_param.vt = VT_BSTR;

/* Transform by specifying the T type element directly */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.9. CaoRobot::Execute("P2T") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"P2T");      //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

//Populate Parameter Option
exe_param.bstrVal = SysAllocString(L"P10");
exe_param.vt = VT_BSTR;

/* Transform P10 value to T type data */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");
```

```
//Populate Parameter Option
exe_param.bstrVal = SysAllocString(L"P(400,400,500,180,0,180,5)");
exe_param.vt = VT_BSTR;

/* Transform by specifying the P type element directly */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.10. CaoRobot::Execute("Dev") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"Dev");      //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

//Populate parameter option
BSTR *param_data;
exe_param.vt = VT_BSTR | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_BSTR, 0, 2);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0] = SysAllocString(L"P10");
param_data[1] = SysAllocString(L"P(100,200,300,180,0,180)");
SafeArrayUnaccessData(exe_param.parray);

/* Calculate the positions of P10 + P(100,200,300,180,0,180) */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.11. CaoRobot::Execute("DevH") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"DevH");     //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

//Populate parameter option
BSTR *param_data;
```

```
exe_param.vt = VT_BSTR | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_BSTR, 0, 2);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0] = SysAllocString(L"P10");
param_data[1] = SysAllocString(L"P(100,200,300,180,0,180)");
SafeArrayUnaccessData(exe_param.parray);

/* Calculate the positions of P10 + Tool Coordinate P(100,200,300,180,0,180) */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.12. CaoRobot::Execute("OutRange") command

Example – Move if the motion range is not exceeded

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"OutRange");  //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

//Populate parameter option
exe_param.bstrVal = SysAllocString(L"P(400,400,300,180,0,180,5)");
exe_param.vt = VT_BSTR;

/* Execute OutRange Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.13. CaoRobot::Execute("MPS") command

Example – Transform an absolute speed to a relative speed

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"MPS");      //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

//Populate parameter option
exe_param.fltVal = 200.0;
exe_param.vt = VT_R4;
```

```
/* Calculate MPS of 200.0 mm/sec */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//TakeArm Command has to be sent before Speed
int32_t speed_axis = -1;                  //Axis Number
float speed_val = exe_result.fltVal;      //Speed Value (%)
hr = bCap_RobotSpeed(fd, hRobot, speed_axis, speed_val);
if (SUCCEEDED(hr))
 printf("bCap_RobotSpeed Succeeded...\n");
else
 printf("bCap_RobotSpeed Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.14. CaoRobot::Execute("RPM") command

Example

```
BSTR exe_comm;                            //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"RPM");        //Command Name
VariantInit(&exe_param);                  //Command Parameters
VariantInit(&exe_result);                 //Command Result

//Populate parameter option
VARIANT *param_data;
exe_param.vt = VT_VARIANT | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_VARIANT, 0, 2);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0].intVal = 1;
param_data[0].vt = VT_I4;
param_data[1].fltVal = 60.0;
param_data[1].vt = VT_R4;
SafeArrayUnaccessData(exe_param.parray);

/* Axis 1, 60.0 RPM */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//TakeArm Command has to be sent before Speed
int32_t speed_axis = -1;                  //Axis Number
float speed_val = exe_result.fltVal;      //Speed Value (%)
hr = bCap_RobotSpeed(fd, hRobot, speed_axis, speed_val);
if (SUCCEEDED(hr))
 printf("bCap_RobotSpeed Succeeded...\n");
else
 printf("bCap_RobotSpeed Failed...\n");
```

```
//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.15. CaoRobot::Execute("DPS") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"DPS");      //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

//Populate parameter option
exe_param.fltVal = 50.0;
exe_param.vt = VT_R4;

/* Move by 50 deg/sec (When in rotation) */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//TakeArm Command has to be sent before Speed
int32_t speed_axis = -1;                //Axis Number
float speed_val = exe_result.fltVal;    //Speed Value (%)
hr = bCap_RobotSpeed(fd, hRobot, speed_axis, speed_val);
if (SUCCEEDED(hr))
 printf("bCap_RobotSpeed Succeeded...\n");
else
 printf("bCap_RobotSpeed Failed...\n");

//Populate parameter option
VARIANT *param_data;
exe_param.vt = VT_VARIANT | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_VARIANT, 0, 2);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0].intVal = 1;
param_data[0].vt = VT_I4;
param_data[1].fltVal = 50.0;
param_data[1].vt = VT_R4;
SafeArrayUnaccessData(exe_param.parray);

/* Move the first axis by 50 deg/sec */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//TakeArm Command has to be sent before Speed
speed_axis = -1;                    //Axis Number
speed_val = exe_result.fltVal;      //Speed Value (%)
hr = bCap_RobotSpeed(fd, hRobot, speed_axis, speed_val);
if (SUCCEEDED(hr))
 printf("bCap_RobotSpeed Succeeded...\n");
```

```
else
 printf("bCap_RobotSpeed Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.16. CaoRobot::Execute("CurPos") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"Curpos");   //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

/* Get Current Position */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.17. CaoRobot::Execute("DestPos") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"DestPos");  //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

/* Get Target Position */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.18. CaoRobot::Execute ("CurPosEx") command

Example

```
BSTR exe_comm;                              //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"CurPosEx");     //Command Name
VariantInit(&exe_param);                    //Command Parameters
```

```
VariantInit(&exe_result);                          //Command Result

/* Timestamp + Get Current Position */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.19. CaoRobot::Execute("DestPosEx") command

Example

```
BSTR exe_comm;                             //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"DestPosEx"); //Command Name
VariantInit(&exe_param);                   //Command Parameters
VariantInit(&exe_result);                  //Command Result

/* Timestamp + Get Target Position */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.20. CaoRobot::Execute("HighCurPosEx") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"HighCurPosEx");     //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

/* Timestamp + Get Current Position */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.21. CaoRobot::Execute("CurJnt") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"CurJnt");   //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

/* Get Current Position */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.22. CaoRobot::Execute("DestJnt") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"DestJnt");  //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

/* Get Target Position */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.23. CaoRobot::Execute("CurJntEx") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"CurJntEx");  //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

/* Timestamp + Get Current Position */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");
```

```
//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.24. CaoRobot::Execute("DestJntEx") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"DestJntEx"); //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

/* Timestamp + Get Target Position */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.25. CaoRobot::Execute("HighCurJntEx") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"HighCurJntEx");     //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

/* Timestamp + Get Current Position */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.26. CaoRobot::Execute("CurTrn") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"CurTrn");   //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result
```

```
/* Get Current Position */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.27. CaoRobot::Execute("DestTrn") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"DestTrn");  //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

/* Get Target Position */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.28. CaoRobot::Execute("CurTrnEx") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"CurTrnEx");  //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

/* Timestamp + Get Current Position (T type) */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.29. CaoRobot::Execute("DestTrnEx") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"DestTrnEx"); //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

/* Timestamp + Get Target Position */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.30. CaoRobot::Execute("HighCurTrnEx") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"HighCurTrnEx");     //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

/* Timestamp + Get Current Position (T Type) */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.31. CaoRobot::Execute("CurFig") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"CurFig");   //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

/* Get Current Figure */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");
```

```
//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.32. CaoRobot::Execute("CurSpd") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"CurSpd");   //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

/* Get Current Speed */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.33. CaoRobot::Execute("CurAcc") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"CurAcc");   //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

/* Get Current Acceleration */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.34. CaoRobot::Execute("CurDec") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"CurDec");   //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result
```

```
/* Get Current Deceleration */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.35. CaoRobot::Execute("CurJSpd") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"CurJSpd");  //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

/* Get Current Joint Speed */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.36. CaoRobot::Execute("CurJAcc") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"CurJAcc");  //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

/* Get Current Joint Accelearation */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.37. CaoRobot::Execute("CurJDec") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"CurJDec");  //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

/* Get Current Joint Deceleration */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.38. CaoRobot::Execute("StartLog") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"StartLog"); //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

/* Start Log */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.39. CaoRobot::Execute("StopLog") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"StopLog");  //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

/* Stop Log */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");
```

```
//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.40. CaoRobot::Execute("ClearLog") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"ClearLog"); //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

/* Clera Log */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.41. CaoRobot::Execute("Motor") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"Motor");    //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

//Populate parameter option
uint32_t *param_data;
exe_param.vt = VT_I4 | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_I4, 0, 2);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0] = 1;
param_data[1] = 0;
SafeArrayUnaccessData(exe_param.parray);

/* Turn on motor and wait for completion of Motor On Process */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.42. CaoRobot::Execute("ExtSpeed") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"ExtSpeed");  //Command Name
VariantInit(&exe_param);                 //Command Parameters
VariantInit(&exe_result);                //Command Result

//Populate parameter option
float *param_data;
exe_param.vt = VT_R4 | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_R4, 0, 3);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0] = 50.0;
param_data[1] = 25.0;
param_data[2] = 25.0;
SafeArrayUnaccessData(exe_param.parray);

/* External Speed = 50%, Acceleration = 25%, Deceleration = 25% */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Populate parameter option
exe_param.fltVal = 50.0;
exe_param.vt = VT_R4;

/* External Speed = 50% (Acceleration, Deceleration are set automatically) */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.43. CaoRobot::Execute("TakeArm") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"Takearm");   //Command Name
VariantInit(&exe_param);                 //Command Parameters
VariantInit(&exe_result);                //Command Result

//Populate parameter option
uint32_t *param_data;
exe_param.vt = VT_I4 | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_I4, 0, 2);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0] = 0;
```

```
param_data[1] = 0;
SafeArrayUnaccessData(exe_param.parray);

/* Init the internal speed to 100, the current tool to 0 and the current work to 0 */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0] = 0;
param_data[1] = 1;
SafeArrayUnaccessData(exe_param.parray);

/* When the internal speed is 50, the current tool is 1, and the current work is 0
   Not initialize the internal speed, current tool and current work, and then get the
   control authority */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.44. CaoRobot::Execute("GiveArm") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"GiveArm");  //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

/* When a program that has executed Takearm is terminated,
   the robot halts before GiveArm is executed
   To terminate the program after completion of Next motion
   explicitly execute GiveArm command */

/* Wait until next motion Completes  Log */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.45. CaoRobot::Execute("Draw") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"Draw");     //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

//Populate parameter option
VARIANT *param_data;
exe_param.vt = VT_VARIANT | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_VARIANT, 0, 2);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0].intVal = 1;                       //Interpolation Method
param_data[0].vt = VT_I4;
param_data[1].bstrVal = SysAllocString(L"V0");  //Distance (Position Data Type)
param_data[1].vt = VT_BSTR;
SafeArrayUnaccessData(exe_param.parray);

/* Draw P, V0 */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

param_data[0].intVal = 2;                               //Interpolation Method
param_data[0].vt = VT_I4;
param_data[1].bstrVal = SysAllocString(L"V(100,100,100)");//Distance (Position Data Type)
param_data[1].vt = VT_BSTR;
SafeArrayUnaccessData(exe_param.parray);

/* Draw L, V(100,100,100) */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.46. CaoRobot::Execute("Approach") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"Approach");  //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

//Populate parameter option
VARIANT *param_data;
exe_param.vt = VT_VARIANT | VT_ARRAY;
```

```
exe_param.parray = SafeArrayCreateVector(VT_VARIANT, 0, 4);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0].intVal = 1;                                   //Interpolation Method
param_data[0].vt = VT_I4;
param_data[1].bstrVal = SysAllocString(L"P1");              //Reference Position
param_data[1].vt = VT_BSTR;
param_data[2].bstrVal = SysAllocString(L"@P 100");          //Approach Length
param_data[2].vt = VT_BSTR;
param_data[3].bstrVal = SysAllocString(L"S=50");            //Motion Option
param_data[3].vt = VT_BSTR;
SafeArrayUnaccessData(exe_param.parray);

/* Approach P, P1, @P 100, S=50 */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

param_data[0].intVal = 2;                                   //Interpolation Method
param_data[0].vt = VT_I4;
param_data[1].bstrVal = SysAllocString(L"P(400,200,350,180,0,180,5)");//Ref Position
param_data[1].vt = VT_BSTR;
param_data[2].bstrVal = SysAllocString(L"@E 56.8");         //Approach Length
param_data[2].vt = VT_BSTR;
param_data[3].bstrVal = SysAllocString(L"S=30, NEXT"); //Motion Option
param_data[3].vt = VT_BSTR;
SafeArrayUnaccessData(exe_param.parray);

/* Approach L, P(400,200,350,180,0,180,5), @P 100, S=30, Next */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.47. CaoRobot::Execute("Depart") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"Depart");   //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

//Populate parameter option
VARIANT *param_data;
exe_param.vt = VT_VARIANT | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_VARIANT, 0, 3);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0].intVal = 1;                                   //Interpolation Method
param_data[0].vt = VT_I4;
param_data[1].bstrVal = SysAllocString(L"@P 100");          //Depart Length
```

```
param_data[1].vt = VT_BSTR;
param_data[2].bstrVal = SysAllocString(L"S=50");       //Motion Option
param_data[2].vt = VT_BSTR;
SafeArrayUnaccessData(exe_param.parray);

/* Depart P, @P 100, S=50 */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

param_data[0].intVal = 2;                              //Interpolation Method
param_data[0].vt = VT_I4;
param_data[1].bstrVal = SysAllocString(L"@E 56.8");    //Depart Length
param_data[1].vt = VT_BSTR;
param_data[2].bstrVal = SysAllocString(L"S=30, NEXT"); //Motion Option
param_data[2].vt = VT_BSTR;
SafeArrayUnaccessData(exe_param.parray);

/* Depart P, @E 56.8, S=30, Next */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.48. CaoRobot::Execute("DriveEx") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"DriveEx");  //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

//Populate parameter option
BSTR *param_data;
exe_param.vt = VT_BSTR | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_BSTR, 0, 2);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0] = SysAllocString(L"@0 (1,10), (2,10)");  //Axis Number and Distance
param_data[1] = SysAllocString(L"S=10, Next");         //Motion Option
SafeArrayUnaccessData(exe_param.parray);

/* DriveEx Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");
```

```
//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.49. CaoRobot::Execute("DriveAEx") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"DriveAEx");  //Command Name
VariantInit(&exe_param);                 //Command Parameters
VariantInit(&exe_result);                //Command Result

//Populate parameter option
BSTR *param_data;
exe_param.vt = VT_BSTR | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_BSTR, 0, 2);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0] = SysAllocString(L"@0 (1,10), (2,10)");  //Axis Number and Distance
param_data[1] = SysAllocString(L"S=10, Next");         //Motion Option
SafeArrayUnaccessData(exe_param.parray);

/* DriveAEx Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.50. CaoRobot::Execute("RotateH") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"RotateH");   //Command Name
VariantInit(&exe_param);                 //Command Parameters
VariantInit(&exe_result);                //Command Result

//Populate parameter option
BSTR *param_data;
exe_param.vt = VT_BSTR | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_BSTR, 0, 2);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0] = SysAllocString(L"@P 32.5");     //Axis Number and Distance
param_data[1] = SysAllocString(L"S=50");        //Motion Option
SafeArrayUnaccessData(exe_param.parray);

/* DriveAEx Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
```

```
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.51. CaoRobot::Execute("Arrive") command

Example

```
BSTR move_opt;                                  //Arguments:
VARIANT move_pos;
uint32_t move_int;
move_int = 1;                                   //Move Interpolation
move_pos.bstrVal = SysAllocString(L"@P P1");    //Move Position
move_pos.vt = VT_BSTR;
move_opt = SysAllocString(L"NEXT");             //Move Option

/* Move P, @P P1, Next */
hr = bCap_RobotMove(fd, hRobot, move_int, move_pos, move_opt);
if (SUCCEEDED(hr))
{
 printf("bCap_RobotMove Succeeded...\n");
 BSTR exe_comm;                                 //Arguments:
 VARIANT exe_param, exe_result;
 exe_comm = SysAllocString(L"Arrive");          //Command Name
 VariantInit(&exe_param);                       //Command Parameters
 VariantInit(&exe_result);                      //Command Result

 //Populate parameter option
 exe_param.fltVal = 50.0;
 exe_param.vt = VT_R4;

 /* Arrive Command */
 hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
 if (SUCCEEDED(hr))
  printf("bCap_RobotExecute Succeeded...\n");
 else
  printf("bCap_RobotExecute Failed...\n");

 //Release Variables
 SysFreeString(exe_comm);
 VariantClear(&exe_param);
 VariantClear(&exe_result);
}
else
 printf("bCap_RobotMove Failed... \n");

//Release Variables
VariantClear(&move_pos);
```

5.2.28.52. CaoRobot::Execute("MotionSkip") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
```

```
exe_comm = SysAllocString(L"MotionSkip");       //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

//Populate parameter option
uint32_t *param_data;
exe_param.vt = VT_I4 | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_I4, 0, 2);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0] = 0;   //Arm group number
param_data[1] = 1;   //Operation Continuation Pattern
SafeArrayUnaccessData(exe_param.parray);

/* DriveAEx Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.53. CaoRobot::Execute("MotionComplete") command

Example

```
BSTR move_opt;                                  //Arguments:
VARIANT move_pos;
uint32_t move_int;
move_int = 1;                                   //Move Interpolation
move_pos.bstrVal = SysAllocString(L"P1");       //Move Position
move_pos.vt = VT_BSTR;
move_opt = SysAllocString(L"NEXT");             //Move Option

/* Move P, P1, Next 'Asyncronous Motion */
hr = bCap_RobotMove(fd, hRobot, move_int, move_pos, move_opt);
if (SUCCEEDED(hr))
{
 printf("bCap_RobotExecute Succeeded...\n");

 BSTR exe_comm;                                 //Arguments:
 VARIANT exe_param, exe_result;
 exe_comm = SysAllocString(L"MotionComplete");  //Command Name
 VariantInit(&exe_param);                       //Command Parameters
 VariantInit(&exe_result);                      //Command Result

 //Populate parameter option
 uint32_t *param_data;
 exe_param.vt = VT_I4 | VT_ARRAY;
 exe_param.parray = SafeArrayCreateVector(VT_I4, 0, 2);
 hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
 param_data[0] = -1; //Arm group number
 param_data[1] = 1;  //Mode
 SafeArrayUnaccessData(exe_param.parray);
```

```
 do
 {
  //Procesing during Motion
  hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
 } while (!exe_result.boolVal);

 //Release Variables
 SysFreeString(exe_comm);
 VariantClear(&exe_param);
 VariantClear(&exe_result);
}
else
 printf("bCab_RobotExecute Failed...\n");

//Release Variables
VariantClear(&move_pos);
```

5.2.28.54. CaoRobot::Execute("CurTool") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"CurTool");  //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

/* CurTool Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.55. CaoRobot::Execute("GetToolDef") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"GetToolDef");       //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

//Populate Parameter
exe_param.intVal = 1;       //Tool Number
exe_param.vt = VT_I4;

/* GetToolDef Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
{
 printf("bCap_RobotExecute Succeeded...\n");
```

```
 //Print Operation Info
 SAFEARRAY* result_list = NULL;
 result_list = V_ARRAY(&exe_result);

 double* result_array;
 hr = SafeArrayAccessData(result_list, (void**)&result_array);
 if (SUCCEEDED(hr))
 {
  printf("Tool Definition: \n");
  printf("X: %f Y: %f Z: %f \n", result_array[0], result_array[1], result_array[2]);
  printf("Rx: %f Ry: %f Rz: %f \n", result_array[3], result_array[4], result_array[5]);

  SafeArrayUnaccessData(result_list);
 }
}
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.56. CaoRobot::Execute("SetToolDef") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"SetToolDef");       //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

//Populate parameter option
VARIANT *param_data;
exe_param.vt = VT_VARIANT | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_VARIANT, 0, 2);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0].intVal = 1;                       //Tool Number
param_data[0].vt = VT_I4;
param_data[1].bstrVal = SysAllocString(L"P2");  //Tool Definition
param_data[1].vt = VT_BSTR;
SafeArrayUnaccessData(exe_param.parray);

/* TOOL 1, P2 */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0].intVal = 2;                                           //Tool Number
param_data[0].vt = VT_I4;
param_data[1].bstrVal = SysAllocString(L"P(100,200,300,180,0,180)"); //Tool Definition
param_data[1].vt = VT_BSTR;
SafeArrayUnaccessData(exe_param.parray);
```

```
/* TOOL 1, P(100,200,300,180,0,180) */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.57. CaoRobot::Execute("CurWork") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"CurWork");  //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

/* CurWork Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.58. CaoRobot::Execute("GetWorkDef") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"GetWorkDef");       //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

//Populate Parameter
exe_param.intVal = 1;       //Work Number
exe_param.vt = VT_I4;

/* GetWorkDef Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
{
 printf("bCap_RobotExecute Succeeded...\n");

 //Print Operation Info
 SAFEARRAY* result_list = NULL;
 result_list = V_ARRAY(&exe_result);

 double* result_array;
```

```
 hr = SafeArrayAccessData(result_list, (void**)&result_array);
 if (SUCCEEDED(hr))
 {
  printf("Work Definition: \n");
  printf("X: %f Y: %f Z: %f \n", result_array[0], result_array[1], result_array[2]);
  printf("Rx: %f Ry: %f Rz: %f \n", result_array[3], result_array[4], result_array[5]);
  printf("ATTR= %f \n", result_array[6]);
  SafeArrayUnaccessData(result_list);
 }
}
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.59. CaoRobot::Execute("SetWorkDef") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"SetWorkDef");       //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

//Populate parameter option
VARIANT *param_data;
exe_param.vt = VT_VARIANT | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_VARIANT, 0, 2);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0].intVal = 1;                       //Tool Number
param_data[0].vt = VT_I4;
param_data[1].bstrVal = SysAllocString(L"P2");  //Tool Definition
param_data[1].vt = VT_BSTR;
SafeArrayUnaccessData(exe_param.parray);

/* Work 1, P2 */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0].intVal = 2;                                           //Tool Number
param_data[0].vt = VT_I4;
param_data[1].bstrVal = SysAllocString(L"P(100,200,300,180,0,180)"); //Tool Definition
param_data[1].vt = VT_BSTR;
SafeArrayUnaccessData(exe_param.parray);

/* Work 1, P(100,200,300,180,0,180) */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");
```

```
//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.60. CaoRobot::Execute("WorkAttribute") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"WorkAttribute");    //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

//Populate Parameter
exe_param.intVal = 1;
exe_param.vt = VT_I4;

/* Work Atrribute Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.61. CaoRobot::Execute("GetAreaDef") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"GetAreaDef");       //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

//Populate Parameter
exe_param.intVal = 1;       //Area Number
exe_param.vt = VT_I4;

/* GetAreaDef Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
{
 printf("bCap_RobotExecute Succeeded...\n");

 //Print Operation Info
 SAFEARRAY* result_list = NULL;
 result_list = V_ARRAY(&exe_result);

 double* result_array;
 hr = SafeArrayAccessData(result_list, (void**)&result_array);
```

```
 if (SUCCEEDED(hr))
 {
  printf("Area Definition: \n");
  printf("X: %f Y: %f Z: %f \n", result_array[0], result_array[1], result_array[2]);
  printf("Rx: %f Ry: %f Rz: %f \n", result_array[3], result_array[4], result_array[5]);
  printf("Dx: %f Dy: %f Dz: %f \n", result_array[6], result_array[7], result_array[8]);
  printf("IOx: %f Error: %f Time: %f \n", result_array[9], result_array[11],
result_array[12]);
  printf("DRx: %f DRy: %f DRz: %f \n", result_array[13], result_array[14],
result_array[15]);
  printf("Position: %f Margin: %f \n", result_array[10], result_array[16]);

  SafeArrayUnaccessData(result_list);
 }
}
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.62. CaoRobot::Execute("SetAreaDef") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"SetAreaDef");       //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

//Populate parameter option
VARIANT *param_data;
exe_param.vt = VT_VARIANT | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_VARIANT, 0, 6);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0].intVal = 1;                       //Area Number
param_data[0].vt = VT_I4;
aram_data[1].bstrVal = SysAllocString(L"P0");   //Position and Rotation
param_data[1].vt = VT_BSTR;
param_data[2].bstrVal = SysAllocString(L"V0");  //Area Size
param_data[2].vt = VT_BSTR;
param_data[3].intVal = 24;                      //I/O number
param_data[3].vt = VT_I4;
param_data[4].intVal = 0;                       //Variable storage number
param_data[4].vt = VT_I4;
param_data[5].intVal = 0;                       //Area detection setting
param_data[5].vt = VT_I4;
SafeArrayUnaccessData(exe_param.parray);

/* Set Area Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");
```

```
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0].intVal = 2;                                    //Area Number
param_data[0].vt = VT_I4;
param_data[1].bstrVal = SysAllocString(L"P(400,250,140,180,0,180)"); //Pos and Rotation
param_data[1].vt = VT_BSTR;
param_data[2].bstrVal = SysAllocString(L"V(200,125,70)");    //Area Size
param_data[2].vt = VT_BSTR;
param_data[3].intVal = 24;                                   //I/O number
param_data[3].vt = VT_I4;
param_data[4].intVal = 0;                                    //Variable storage number
param_data[4].vt = VT_I4;
param_data[5].intVal = 0;                                    //Area detection setting
param_data[5].vt = VT_I4;
SafeArrayUnaccessData(exe_param.parray);

/* Set Area Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.63. CaoRobot::Execute("SetArea") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"SetArea");  //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

//Populate Parameter
exe_param.intVal = 1;       //Area Number
exe_param.vt = VT_I4;

/* Set Area Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.64. CaoRobot::Execute("ResetArea") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
```

```
exe_comm = SysAllocString(L"ResetArea");  //Command Name
VariantInit(&exe_param);                  //Command Parameters
VariantInit(&exe_result);                 //Command Result

//Populate Parameter
exe_param.intVal = 1;       //Area Number
exe_param.vt = VT_I4;

/* Reset Area Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.65. CaoRobot::Execute("AreaSize") command

Example

```
BSTR exe_comm;                            //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"AreaSize");   //Command Name
VariantInit(&exe_param);                  //Command Parameters
VariantInit(&exe_result);                 //Command Result

//Populate Parameter
exe_param.intVal = 1;       //Area Number
exe_param.vt = VT_I4;

/* Area Size Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
{
 printf("bCap_RobotExecute Succeeded...\n");
 //Print Operation Info
 SAFEARRAY* result_list = NULL;
 result_list = V_ARRAY(&exe_result);

 double* result_array;
 hr = SafeArrayAccessData(result_list, (void**)&result_array);
 if (SUCCEEDED(hr))
 {
  printf("Area Size: \n");
  printf("X: %f Y: %f Z: %f \n", result_array[0], result_array[1], result_array[2]);
  SafeArrayUnaccessData(result_list);
 }
}
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.66. CaoRobot::Execute("GetAreaEnabled") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"GetAreaEnabled");   //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

//Populate Parameter
exe_param.intVal = 1;       //Area Number
exe_param.vt = VT_I4;

/* GetAreaEnabled Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.67. CaoRobot::Execute("SetAreaEnabled") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"SetAreaEnabled");   //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

//Populate parameter option
VARIANT *param_data;
exe_param.vt = VT_VARIANT | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_VARIANT, 0, 2);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0].intVal = 1;          //Area Number
param_data[0].vt = VT_I4;
param_data[1].boolVal = 1;         //Enable/Disable Option
param_data[1].vt = VT_BOOL;
SafeArrayUnaccessData(exe_param.parray);

/* Set Area 1 Enabled */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.68. CaoRobot::Execute("AddPathPoint") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"AddPathPoint");     //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

//Populate parameter option
VARIANT *param_data;
exe_param.vt = VT_VARIANT | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_VARIANT, 0, 2);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0].intVal = 2;                                           //Path Number
param_data[0].vt = VT_I4;
param_data[1].bstrVal = SysAllocString(L"P(400,200,140,180,0,180)"); //PoseData
param_data[1].vt = VT_BSTR;
SafeArrayUnaccessData(exe_param.parray);

/* AddPathPoint 2, P(400,200,140,180,0,180) */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.69. CaoRobot::Execute("ClrPathPoint") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"ClrPathPoint");     //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

//Populate parameter option
exe_param.intVal = 2;       //Path Number
exe_param.vt = VT_I4;

/* AddPathPoint 2, P(400,200,140,180,0,180) */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.70. CaoRobot::Execute("GetPathPoint") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"GetPathPoint");     //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

//Populate parameter option
uint32_t *param_data;
exe_param.vt = VT_I4 | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_I4, 0, 2);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0] = 2;   //Path Number
param_data[1] = 1;   //Path Point number
SafeArrayUnaccessData(exe_param.parray);

/* GetPathPoint Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.71. CaoRobot::Execute("LoadPathPoint") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"LoadPathPoint");    //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

//Populate parameter option
exe_param.intVal = 2;       //Path Number
exe_param.vt = VT_I4;

/* LoadPathPoint Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.72. CaoRobot::Execute("GetPathPointCount ") command

Example

```
BSTR exe_comm;                                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"GetPathPointCount");        //Command Name
VariantInit(&exe_param);                                //Command Parameters
VariantInit(&exe_result);                               //Command Result

//Populate parameter option
exe_param.intVal = 2;       //Path Number
exe_param.vt = VT_I4;

/* LoadPathPoint Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.73. CaoRobot::Execute("GetRobotTypeName") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"GetRobotTypeName"); //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

/* Get Robot Type Name Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.74. CaoRobot::Execute("ArchMove") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"ArchMove"); //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

//Populate parameter option
VARIANT *param_data;
```

```
exe_param.vt = VT_VARIANT | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_VARIANT, 0, 4);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0].bstrVal = SysAllocString(L"P10"); //Target Position
param_data[0].vt = VT_BSTR;
param_data[1].fltVal = 50.0;                    //Height
param_data[1].vt = VT_R4;
param_data[2].fltVal = 30.0;                    //Arch Start Position
param_data[2].vt = VT_R4;
param_data[3].fltVal = 30.0;                    //Arch Stop Position
param_data[3].vt = VT_R4;
SafeArrayUnaccessData(exe_param.parray);

/* ArchMove Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.75. CaoRobot::Execute("CrtMotionAllow") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"CrtMotionAllow");   //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

//Populate parameter option
VARIANT *param_data;
exe_param.vt = VT_VARIANT | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_VARIANT, 0, 3);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0].intVal = 1;   //Set/Reset Option
param_data[0].vt = VT_I4;
param_data[1].fltVal = 1.0; //Positional precision (mm)
param_data[1].vt = VT_R4;
param_data[2].fltVal = 1.0; //Postural precision (deg)
param_data[2].vt = VT_R4;
SafeArrayUnaccessData(exe_param.parray);

/* CrtMotionAllow ON Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

BSTR move_opt;                                  //Arguments:
VARIANT move_pos;
uint32_t move_int;
move_int = 1;                                   //Move Interpolation
```

```
move_pos.bstrVal = SysAllocString(L"@C J2");     //Move Position
move_pos.vt = VT_BSTR;
move_opt = SysAllocString(L"");                  //Move Option

/* Start Robot Motion */
hr = bCap_RobotMove(fd, hRobot, move_int, move_pos, move_opt);
if (SUCCEEDED(hr))
 printf("bCap_RobotMove Succeeded...\n");
else
 printf("bCap_RobotMove Failed... \n");

//Populate parameter option
exe_param.bstrVal = 0;
exe_param.vt = VT_I4;

/* CrtMotionAllow OFF Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.76. CaoRobot::Execute("EncMotionAllow") command

Example

```
BSTR exe_comm;                                   //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"EncMotionAllow");    //Command Name
VariantInit(&exe_param);                         //Command Parameters
VariantInit(&exe_result);                        //Command Result

//Populate parameter option
VARIANT *param_data;
exe_param.vt = VT_VARIANT | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_VARIANT, 0, 3);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0].intVal = 1;   //Set/Reset Option
param_data[0].vt = VT_I4;
param_data[1].fltVal = 1.0; //Positional precision (mm)
param_data[1].vt = VT_R4;
param_data[2].fltVal = 1.0; //Postural precision (deg)
param_data[2].vt = VT_R4;
SafeArrayUnaccessData(exe_param.parray);

/* EncMotionAllow ON Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

BSTR move_opt;                                   //Arguments:
VARIANT move_pos;
```

```
uint32_t move_int;
move_int = 1;                                   //Move Interpolation
move_pos.bstrVal = SysAllocString(L"@E J2");    //Move Position
move_pos.vt = VT_BSTR;
move_opt = SysAllocString(L"");                 //Move Option

/* Start Robot Motion */
hr = bCap_RobotMove(fd, hRobot, move_int, move_pos, move_opt);
if (SUCCEEDED(hr))
 printf("bCap_RobotMove Succeeded...\n");
else
 printf("bCap_RobotMove Failed... \n");

//Populate parameter option
exe_param.bstrVal = 0;
exe_param.vt = VT_BOOL;

/* EncMotionAllow OFF Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.77. CaoRobot::Execute("EncMotionAllowJnt") command

Example

```
BSTR exe_comm;                                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"EncMotionAllowJnt");        //Command Name
VariantInit(&exe_param);                                //Command Parameters
VariantInit(&exe_result);                               //Command Result

//Populate parameter option
VARIANT *param_data;
exe_param.vt = VT_VARIANT | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_VARIANT, 0, 4);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0].intVal = 1;          //Set/Reset Option
param_data[0].vt = VT_I4;
param_data[1].fltVal = 7;          //Axis Number
param_data[1].vt = VT_R4;
param_data[2].fltVal = 0.01;       //Allowable angle
param_data[2].vt = VT_R4;
param_data[3].fltVal = 1.0;        //Mode Value
param_data[3].vt = VT_I4;
SafeArrayUnaccessData(exe_param.parray);

/* EncMotionAllowJnt ON Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
```

```
else
  printf("bCap_RobotExecute Failed...\n");

BSTR move_opt;                                                  //Arguments:
VARIANT move_pos;
uint32_t move_int;
move_int = 1;                                                   //Move Interpolation
move_pos.bstrVal = SysAllocString(L"@E J2 EXA(7,30.5)");        //Move Position
move_pos.vt = VT_BSTR;
move_opt = SysAllocString(L"");                                 //Move Option

/* Start Robot Motion */
hr = bCap_RobotMove(fd, hRobot, move_int, move_pos, move_opt);
if (SUCCEEDED(hr))
 printf("bCap_RobotMove Succeeded...\n");
else
 printf("bCap_RobotMove Failed... \n");

//Populate parameter option
exe_param.parray = SafeArrayCreateVector(VT_VARIANT, 0, 2);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0].intVal = 0;          //Set/Reset Option
param_data[0].vt = VT_I4;
param_data[1].fltVal = 7;          //Axis Number
param_data[1].vt = VT_R4;
SafeArrayUnaccessData(exe_param.parray);

/* EncMotionAllowJnt OFF Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.78. CaoRobot::Execute("ErAlw") command

Example

```
//Example 1-----------------------------------------------------------
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"ErAlw");    //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

//Populate parameter option
VARIANT *param_data;
exe_param.vt = VT_VARIANT | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_VARIANT, 0, 3);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0].intVal = 1;          //Set/Reset
param_data[0].vt = VT_I4;
param_data[1].intVal = 1;          //Axis Number
param_data[1].vt = VT_I4;
```

```
param_data[2].fltVal = 0.01;       //Setting Value (deg or mm)
param_data[2].vt = VT_R4;
SafeArrayUnaccessData(exe_param.parray);

/* ErAlw Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Example 2-----------------------------------------------------------
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0].intVal = 1;          //Set/Reset
param_data[0].vt = VT_I4;
param_data[1].intVal = 2;          //Axis Number
param_data[1].vt = VT_I4;
param_data[2].fltVal = 0.01;       //Setting Value (deg or mm)
param_data[2].vt = VT_R4;
SafeArrayUnaccessData(exe_param.parray);

/* ErAlw Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Example 3-----------------------------------------------------------
exe_param.parray = SafeArrayCreateVector(VT_VARIANT, 0, 2);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0].intVal = 0;          //Set/Reset
param_data[0].vt = VT_I4;
param_data[1].intVal = 0;          //Axis Number
param_data[1].vt = VT_I4;
SafeArrayUnaccessData(exe_param.parray);

/* ErAlw Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.79. CaoRobot::Execute("ForceCtrl") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"ForceCtrl"); //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result
```

```
//Populate parameter option
uint32_t *param_data;
exe_param.vt = VT_I4 | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_I4, 0, 2);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0] = 1;   //Set/Reset Option
param_data[1] = 1;   //Force Control Number
SafeArrayUnaccessData(exe_param.parray);

/* ForceCtrl ON Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Populate parameter option
exe_param.intVal = 0;       //Set/Reset Option
exe_param.vt = VT_I4;

/* ForceCtrl OFF Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.80. CaoRobot::Execute("ForceParam") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"ForceParam");       //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

//Populate parameter option
VARIANT *param_data;
exe_param.vt = VT_VARIANT | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_VARIANT, 0, 3);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0].intVal = 1;                       //Set/Reset Option
param_data[0].vt = VT_I4;
param_data[1].intVal = 1;                       //Coordinates
param_data[1].vt = VT_I4;
param_data[2].bstrVal = SysAllocString(L"P10"); //Force
param_data[2].vt = VT_BSTR;
SafeArrayUnaccessData(exe_param.parray);

/* ForceParam Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
```

```
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.81. CaoRobot::Execute("ForceValue") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"ForceValue");       //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

//Populate parameter option
uint32_t *param_data;
exe_param.vt = VT_I4 | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_I4, 0, 2);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0] = 1;   //Data Number
param_data[1] = 0;   //Mode
SafeArrayUnaccessData(exe_param.parray);

/* ForceValue Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.82. CaoRobot::Execute("ForceWaitCondition") command

Example

```
BSTR exe_comm;                                      //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"ForceWaitCondition");   //Command Name
VariantInit(&exe_param);                            //Command Parameters
VariantInit(&exe_result);                           //Command Result

//Populate parameter option
BSTR *param_data;
exe_param.vt = VT_BSTR | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_BSTR, 0, 2);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0] = SysAllocString(L"P0");    //Position
param_data[1] = SysAllocString(L"P1");    //Force
SafeArrayUnaccessData(exe_param.parray);
```

```
/* ForceWaitCondition Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.83. CaoRobot::Execute("ForceSensor") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"ForceSensor");      //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

//Populate parameter option
exe_param.vt = VT_I4;
exe_param.intVal = 0;       //Set/Reset Option

/* ForceSensor Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.84. CaoRobot::Execute("ForceChangeTable") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"ForceChangeTable"); //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

//Populate parameter option
exe_param.vt = VT_I4;
exe_param.intVal = 1;       //Table Number

/* ForceChangeTable Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");
```

```
//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.85. CaoRobot::Execute("GetSrvData") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"GetSrvData");       //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

//Populate parameter option
exe_param.vt = VT_I4;
exe_param.intVal = 2;       //Data Number

/* GetSrvData Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.86. CaoRobot::Execute("GetSrvJntData") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"GetSrvJntData");    //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

//Populate parameter option
uint32_t *param_data;
exe_param.vt = VT_I4 | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_I4, 0, 2);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
aram_data[0] = 2;    //Data Number
param_data[1] = 1;   //Axis Number
SafeArrayUnaccessData(exe_param.parray);

/* GetSrvJntData Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
```

```
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.87. CaoRobot::Execute("GrvCtrl") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"GrvCtrl");  //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

//Populate parameter option
exe_param.vt = VT_I4;
exe_param.intVal = 1;       //Set/Reset Option

/* GrvCtrl ON Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Populate parameter option
exe_param.vt = VT_I4;
exe_param.intVal = 0;       //Set/Reset Option

/* GrvCtrl OFF Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.88. CaoRobot::Execute("CurLmt") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"GrvCtrl");  //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

//Populate parameter option
exe_param.vt = VT_I4;
exe_param.intVal = 1;       //Set/Reset Option

/* GrvCtrl ON Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
```

```
else
 printf("bCap_RobotExecute Failed...\n");

exe_comm = SysAllocString(L"CurLmt");     //Command Name

//Populate parameter option
VARIANT *param_data;
exe_param.vt = VT_VARIANT | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_VARIANT, 0, 3);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0].intVal = 1;          //Set Reset
param_data[0].vt = VT_I4;
param_data[1].fltVal = 1;          //Axis Number
param_data[1].vt = VT_R4;
param_data[2].fltVal = 10.5;       //Setting Value
param_data[2].vt = VT_R4;
SafeArrayUnaccessData(exe_param.parray);

/* CurLmt(On, 1, 10.5) Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Populate parameter option
exe_param.vt = VT_VARIANT | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_VARIANT, 0, 3);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0].intVal = 1;          //Set Reset
param_data[0].vt = VT_I4;
param_data[1].fltVal = 2;          //Axis Number
param_data[1].vt = VT_R4;
param_data[2].fltVal = 50.3;       //Setting Value
param_data[2].vt = VT_R4;
SafeArrayUnaccessData(exe_param.parray);

/* CurLmt(On, 2, 50.3) Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Populate parameter option
exe_param.vt = VT_VARIANT | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_VARIANT, 0, 2);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0].intVal = 0;   //Set Reset
param_data[0].vt = VT_I4;
param_data[1].fltVal = 0;   //Axis Number
param_data[1].vt = VT_R4;
SafeArrayUnaccessData(exe_param.parray);

/* CurLmt(Off, 0) Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
```

```
else
 printf("bCap_RobotExecute Failed...\n");

exe_comm = SysAllocString(L"GrvCtrl");    //Command Name

//Populate parameter option
exe_param.vt = VT_I4;
exe_param.intVal = 0;       //Set/Reset Option

/* GrvCtrl OFF Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.89. CaoRobot::Execute("Zforce") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"GrvCtrl");    //Command Name
VariantInit(&exe_param);                  //Command Parameters
VariantInit(&exe_result);                 //Command Result

//Populate parameter option
exe_param.vt = VT_I4;
exe_param.intVal = 1;       //Set/Reset Option

/* GrvCtrl ON Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

exe_comm = SysAllocString(L"Zforce");     //Command Name

//Populate parameter option
exe_param.vt = VT_R4;
exe_param.intVal = 50.0;    //Thrust Force

/* Zforce Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

exe_comm = SysAllocString(L"GrvCtrl");    //Command Name

//Populate parameter option
exe_param.vt = VT_I4;
```

```
exe_param.intVal = 0;       //Set/Reset Option

/* GrvCtrl OFF Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.90. CaoRobot::Execute("GrvOffset") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"GrvOffset"); //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

//Populate parameter option
exe_param.vt = VT_I4;
exe_param.intVal = 1;       //Set/Reset Option

/* GrvOffset ON Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Populate parameter option
exe_param.vt = VT_I4;
exe_param.intVal = 0;       //Set/Reset Option

/* GrvOffset OFF Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.91. CaoRobot::Execute("HighPathAccuracy") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"HighPathAccuracy"); //Command Name
VariantInit(&exe_param);                        //Command Parameters
```

```
VariantInit(&exe_result);                          //Command Result

//Populate parameter option
exe_param.vt = VT_I4;
exe_param.intVal = 1;       //Set/Reset Option

/* HighPathAccuracy ON Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Populate parameter option
exe_param.vt = VT_I4;
exe_param.intVal = 0;       //Set/Reset Option

/* HighPathAccuracy OFF Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.92. CaoRobot::Execute("MotionTimeout") command

Example

```
BSTR exe_comm;                                     //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"MotionTimeOut");     //Command Name
VariantInit(&exe_param);                           //Command Parameters
VariantInit(&exe_result);                          //Command Result

//Populate parameter option
uint32_t *param_data;
exe_param.vt = VT_I4 | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_I4, 0, 2);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0] = 1;          //Set/Reset
param_data[1] = 1000;       //Timeout Period
SafeArrayUnaccessData(exe_param.parray);

/* MotionTimeout Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.93. CaoRobot::Execute("SingularAvoid") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"SingularAvoid");    //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

//Populate parameter option
exe_param.vt = VT_I4;
exe_param.intVal = 2;       //Mode

/* SingularAvoid ON Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

BSTR move_opt;                                  //Arguments:
VARIANT move_pos;
uint32_t move_int;
move_int = 1;                                   //Move Interpolation
move_pos.bstrVal = SysAllocString(L"@0 P2");    //Move Position
move_pos.vt = VT_BSTR;
move_opt = SysAllocString(L"");                 //Move Option

/* Move P, @0 P2 Command */
hr = bCap_RobotMove(fd, hRobot, move_int, move_pos, move_opt);
if (SUCCEEDED(hr))
 printf("bCap_RobotMove Succeeded...\n");
else
 printf("bCap_RobotMove Failed... \n");

//Populate parameter option
exe_param.vt = VT_I4;
exe_param.intVal = 0;       //Mode

/* SingularAvoid OFF Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
SysFreeString(move_opt);
VariantClear(&exe_param);
VariantClear(&exe_result);
VariantClear(&move_pos);
```

5.2.28.94. CaoRobot::Execute("SpeedMode") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"SpeedMode"); //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

//Populate parameter option
exe_param.vt = VT_I4;
exe_param.intVal = 1;       //Mode Number

/* SpeedMode Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.95. CaoRobot::Execute("PayLoad") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"PayLoad");  //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

//Populate parameter option
VARIANT *param_data;
exe_param.vt = VT_VARIANT | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_VARIANT, 0, 3);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0].intVal = 2000;                          //Payload
param_data[0].vt = VT_I4;
param_data[1].bstrVal = SysAllocString(L"V(0,100,150)");//Payload Center of Gravity
param_data[1].vt = VT_BSTR;
param_data[2].bstrVal = SysAllocString(L"V(0,10,10)");//Payload Center of Gravity Inertia
param_data[2].vt = VT_BSTR;
SafeArrayUnaccessData(exe_param.parray);

/* PayLoad Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
```

```
VariantClear(&exe_result);
```

5.2.28.96. CaoRobot::Execute("GenerateNonStopPath ") command

Example

```
       //Hard to replicate example on RC8 Provider
       //May need to do additional Testing
```

5.2.28.97. CaoRobot::Execute("RobInfo") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"RobInfo");  //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

//Populate parameter option
exe_param.vt = VT_I4;
exe_param.intVal = 0;       //Index Number

/* RobInfo Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.98. CaoRobot::Execute("SyncTimeStart") command

Example

```
//Get Robot Handler
BSTR rob_name, rob_opt;                 //Arguments:
rob_name = SysAllocString(L"Robot0");   //Name
rob_opt = SysAllocString(L"ID=0");      //Option

/* Obtain Robot0 Reference */
hr = bCap_ControllerGetRobot(fd, hCtrl, rob_name, rob_opt, &hRobot0);
if (SUCCEEDED(hr))
 printf("bCap_ControllerGetRobot Succeeded...\n");
else
 printf("bCap_ControllerGetRobot Failed...\n");

rob_name = SysAllocString(L"Robot1");   //Name
rob_opt = SysAllocString(L"ID=1");      //Option

/* Obtain Robot1 Reference */
hr = bCap_ControllerGetRobot(fd, hCtrl, rob_name, rob_opt, &hRobot1);
if (SUCCEEDED(hr))
 printf("bCap_ControllerGetRobot Succeeded...\n");
else
 printf("bCap_ControllerGetRobot Failed...\n");
```

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"SyncTimeStart");    //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

/* SyncStart Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

BSTR move_opt;                                  //Arguments:
VARIANT move_pos;
uint32_t move_int;
move_int = 1;                                   //Move Interpolation
move_pos.bstrVal = SysAllocString(L"P1");       //Move Position
move_pos.vt = VT_BSTR;
move_opt = SysAllocString(L"");                 //Move Option

/* Instruct that Master Robot to Move to P1 */
hr = bCap_RobotMove(fd, hRobot0, move_int, move_pos, move_opt);
if (SUCCEEDED(hr))
 printf("bCap_RobotMove Succeeded...\n");
else
 printf("bCap_RobotMove Failed... \n");

move_int = 2;                                   //Move Interpolation
move_pos.bstrVal = SysAllocString(L"P3");       //Move Position
move_pos.vt = VT_BSTR;
move_opt = SysAllocString(L"");                 //Move Option

/* Instruct that Slave Robot to Move to P3 */
hr = bCap_RobotMove(fd, hRobot1, move_int, move_pos, move_opt);
if (SUCCEEDED(hr))
 printf("bCap_RobotMove Succeeded...\n");
else
 printf("bCap_RobotMove Failed... \n");

exe_comm = SysAllocString(L"SyncTimeEnd");      //Command Name

/* SyncEnd Command: Start synchronous motion */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(rob_name);
SysFreeString(rob_opt);
SysFreeString(exe_comm);
SysFreeString(move_opt);
VariantClear(&exe_param);
VariantClear(&exe_result);
VariantClear(&move_pos);
```

5.2.28.99. CaoRobot::Execute("SyncTimeEnd") command

Example

```
//Get Robot Handler
BSTR rob_name, rob_opt;                    //Arguments:
rob_name = SysAllocString(L"Robot0");      //Name
rob_opt = SysAllocString(L"ID=0");         //Option

/* Obtain Robot0 Reference */
hr = bCap_ControllerGetRobot(fd, hCtrl, rob_name, rob_opt, &hRobot0);
if (SUCCEEDED(hr))
       printf("bCap_ControllerGetRobot Succeeded...\n");
else
       printf("bCap_ControllerGetRobot Failed...\n");

rob_name = SysAllocString(L"Robot1");      //Name
rob_opt = SysAllocString(L"ID=1");         //Option

/* Obtain Robot1 Reference */
hr = bCap_ControllerGetRobot(fd, hCtrl, rob_name, rob_opt, &hRobot1);
if (SUCCEEDED(hr))
 printf("bCap_ControllerGetRobot Succeeded...\n");
else
 printf("bCap_ControllerGetRobot Failed...\n");

BSTR exe_comm;                                    //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"SyncTimeStart");      //Command Name
VariantInit(&exe_param);                          //Command Parameters
VariantInit(&exe_result);                         //Command Result

/* SyncStart Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

BSTR move_opt;                                    //Arguments:
VARIANT move_pos;
uint32_t move_int;
move_int = 1;                                     //Move Interpolation
move_pos.bstrVal = SysAllocString(L"P1");         //Move Position
move_pos.vt = VT_BSTR;
move_opt = SysAllocString(L"");                   //Move Option

/* Instruct that Master Robot to Move to P1 */
hr = bCap_RobotMove(fd, hRobot0, move_int, move_pos, move_opt);
if (SUCCEEDED(hr))
 printf("bCap_RobotMove Succeeded...\n");
else
 printf("bCap_RobotMove Failed... \n");

move_int = 1;                                     //Move Interpolation
move_pos.bstrVal = SysAllocString(L"P3");         //Move Position
move_pos.vt = VT_BSTR;
move_opt = SysAllocString(L"");                   //Move Option
```

```
/* Instruct that Slave Robot to Move to P3 */
hr = bCap_RobotMove(fd, hRobot1, move_int, move_pos, move_opt);
if (SUCCEEDED(hr))
 printf("bCap_RobotMove Succeeded...\n");
else
 printf("bCap_RobotMove Failed... \n");

exe_comm = SysAllocString(L"SyncTimeEnd");      //Command Name

//Populate parameter option
exe_param.vt = VT_I4;
exe_param.intVal = 0;       //Motion Option

/* SyncEnd Command: Start synchronous motion with Next Option */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(rob_name);
SysFreeString(rob_opt);
SysFreeString(exe_comm);
SysFreeString(move_opt);
VariantClear(&exe_param);
VariantClear(&exe_result);
VariantClear(&move_pos);
```

5.2.28.100. CaoRobot::Execute("SyncMoveStart") command

Example

```
//Get Robot Handler
BSTR rob_name, rob_opt;                  //Arguments:
rob_name = SysAllocString(L"Robot0");    //Name
rob_opt = SysAllocString(L"ID=0");       //Option

/* Obtain Robot0 Reference */
hr = bCap_ControllerGetRobot(fd, hCtrl, rob_name, rob_opt, &hRobot0);
if (SUCCEEDED(hr))
 printf("bCap_ControllerGetRobot Succeeded...\n");
else
 printf("bCap_ControllerGetRobot Failed...\n");

rob_name = SysAllocString(L"Robot1");    //Name
rob_opt = SysAllocString(L"ID=1");       //Option

/* Obtain Robot1 Reference */
hr = bCap_ControllerGetRobot(fd, hCtrl, rob_name, rob_opt, &hRobot1);
if (SUCCEEDED(hr))
 printf("bCap_ControllerGetRobot Succeeded...\n");
else
 printf("bCap_ControllerGetRobot Failed...\n");

BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"SyncMoveStart");    //Command Name
VariantInit(&exe_param);                        //Command Parameters
```

```
VariantInit(&exe_result);                          //Command Result

//Populate parameter option
uint32_t *param_data;
exe_param.vt = VT_I4 | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_I4, 0, 1);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0] = 1;          //RobotID of the follower Robot
SafeArrayUnaccessData(exe_param.parray);

/* SyncMoveStart Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

BSTR move_opt;                                                  //Arguments:
VARIANT move_pos;
uint32_t move_int;
move_int = 2;                                                   //Move Interpolation
move_pos.bstrVal = SysAllocString(L"J(0,45,90,0,45,0,0,0)");  //Move Position
move_pos.vt = VT_BSTR;
move_opt = SysAllocString(L"");                                 //Move Option

/* Instruct the leader robot to move to the specified postion */
hr = bCap_RobotMove(fd, hRobot0, move_int, move_pos, move_opt);
if (SUCCEEDED(hr))
 printf("bCap_RobotMove Succeeded...\n");
else
 printf("bCap_RobotMove Failed... \n");

exe_comm = SysAllocString(L"SyncMoveEnd");      //Command Name
VariantClear(&exe_param);                       //Clears Parameter Variant

/* Start the cooperative motion of Robot0 and Robot 1 */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(rob_name);
SysFreeString(rob_opt);
SysFreeString(exe_comm);
SysFreeString(move_opt);
VariantClear(&exe_param);
VariantClear(&exe_result);
VariantClear(&move_pos);
```

5.2.28.101. CaoRobot::Execute("SyncMoveEnd") command

Example

```
//Get Robot Handler
BSTR rob_name, rob_opt;                  //Arguments:
rob_name = SysAllocString(L"Robot0");    //Name
rob_opt = SysAllocString(L"ID=0");       //Option
```

```
/* Obtain Robot0 Reference */
hr = bCap_ControllerGetRobot(fd, hCtrl, rob_name, rob_opt, &hRobot0);
if (SUCCEEDED(hr))
 printf("bCap_ControllerGetRobot Succeeded...\n");
else
 printf("bCap_ControllerGetRobot Failed...\n");

rob_name = SysAllocString(L"Robot1");     //Name
rob_opt = SysAllocString(L"ID=1");        //Option

/* Obtain Robot1 Reference */
hr = bCap_ControllerGetRobot(fd, hCtrl, rob_name, rob_opt, &hRobot1);
if (SUCCEEDED(hr))
 printf("bCap_ControllerGetRobot Succeeded...\n");
else
 printf("bCap_ControllerGetRobot Failed...\n");

BSTR exe_comm;                                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"SpeedMode");                //Command Name
VariantInit(&exe_param);                                //Command Parameters
VariantInit(&exe_result);                               //Command Result

//Populate parameter option
exe_param.vt = VT_I4;
exe_param.intVal = 1;       //Mode Number

/* SpeedMode Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

exe_comm = SysAllocString(L"SyncMoveStart");    //Command Name

//Populate parameter option
uint32_t *param_data;
exe_param.vt = VT_I4 | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_I4, 0, 1);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0] = 1;          //RobotID of the follower Robot
SafeArrayUnaccessData(exe_param.parray);

/* SyncMoveStart Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

BSTR move_opt;                                              //Arguments:
VARIANT move_pos;
uint32_t move_int;
move_int = 2;                                               //Move Interpolation
move_pos.bstrVal = SysAllocString(L"J(0,45,90,0,45,0,0,0)"); //Move Position
move_pos.vt = VT_BSTR;
```

```
move_opt = SysAllocString(L"");                                  //Move Option

/* Instruct the leader robot to move to the specified postion */
hr = bCap_RobotMove(fd, hRobot0, move_int, move_pos, move_opt);
if (SUCCEEDED(hr))
 printf("bCap_RobotMove Succeeded...\n");
else
 printf("bCap_RobotMove Failed... \n");

exe_comm = SysAllocString(L"SyncMoveEnd");       //Command Name

//Populate parameter option
exe_param.vt = VT_I4;
exe_param.intVal = 1;

/* Start the cooperative motion of Robot0 and Robot 1 with next option */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(rob_name);
SysFreeString(rob_opt);
SysFreeString(exe_comm);
SysFreeString(move_opt);
VariantClear(&exe_param);
VariantClear(&exe_result);
VariantClear(&move_pos);
```

5.2.28.102. CaoRobot::Execute("SetBaseDef") command

Example

```
BSTR exe_comm;                              //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"SetBaseDef"); //Command Name
VariantInit(&exe_param);                    //Command Parameters
VariantInit(&exe_result);                   //Command Result

//Populate parameter option
VARIANT *param_data;
exe_param.vt = VT_VARIANT | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_VARIANT, 0, 2);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0].intVal = 1;                           //Base Number
param_data[0].vt = VT_I4;
param_data[1].bstrVal = SysAllocString(L"P2");   //Base Definition
param_data[1].vt = VT_BSTR;
SafeArrayUnaccessData(exe_param.parray);

/* SetBaseDef Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");
```

```
exe_comm = SysAllocString(L"SetWorkDef"); //Command Name

//Populate parameter option
exe_param.vt = VT_VARIANT | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_VARIANT, 0, 2);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0].intVal = 1;                                          //Work Number
param_data[0].vt = VT_I4;
param_data[1].bstrVal = SysAllocString(L"P(100,200,300,180,0,180)"); //Work Definition
param_data[1].vt = VT_BSTR;
SafeArrayUnaccessData(exe_param.parray);

/* SetWorkDef Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.103. CaoRobot::Execute("GetBaseDef") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"GetBaseDef"); //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

//Populate parameter option
exe_param.vt = VT_I4;
exe_param.intVal = 1;       //Base Number

/* GetBaseDef Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.104. CaoRobot::Execute("SetHandIO") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"SetHandIO");  //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result
```

```
//Populate parameter option
uint32_t *param_data;
exe_param.vt = VT_I4 | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_I4, 0, 3);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0] = 64;         //The smallest Hand I/O number
param_data[1] = 8;          //Values to be set
param_data[2] = 4;          //Setting Range
SafeArrayUnaccessData(exe_param.parray);

/* SetHandIO Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.105. CaoRobot::Execute("GetHandIO") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"GetHandIO"); //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

//Populate parameter option
uint32_t *param_data;
exe_param.vt = VT_I4 | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_I4, 0, 2);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0] = 48;         //The smallest Hand I/O number
param_data[1] = 4;          //Setting Range
SafeArrayUnaccessData(exe_param.parray);

/* GetHandIO Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.106. CaoRobot::Execute("StartServoLog") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"StartServoLog");    //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

/* StartServoLog Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.107. CaoRobot::Execute("ClearServoLog") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"ClearServoLog");    //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

/* ClearServoLog Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.108. CaoRobot::Execute("StopServoLog") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"StopServoLog");     //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

/* StopServoLog Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");
```

```
//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.109. CaoRobot::Execute("GetCtrlLogMaxTime") command

Example

```
BSTR exe_comm;                                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"GetCtrlLogMaxTime");        //Command Name
VariantInit(&exe_param);                                //Command Parameters
VariantInit(&exe_result);                               //Command Result

/* GetCtrlLogMaxTime Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.110. CaoRobot::Execute("SetCtrlLogMaxTime") command

Example

```
BSTR exe_comm;                                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"SetCtrlLogMaxTime");        //Command Name
VariantInit(&exe_param);                                //Command Parameters
VariantInit(&exe_result);                               //Command Result

//Populate parameter option
exe_param.vt = VT_I4;
exe_param.intVal = 10;      //Logging duration to be set

/* SetCtrlLogMaxTime Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.111. CaoRobot::Execute("GetCtrlLogInterval") command

Example

```
BSTR exe_comm;                                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"GetCtrlLogInterval");       //Command Name
VariantInit(&exe_param);                                //Command Parameters
VariantInit(&exe_result);                               //Command Result

/* GetCtrlLogInterval Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.112. CaoRobot::Execute("SetCtrlLogInterval ") command

Example

```
BSTR exe_comm;                                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"SetCtrlLogInterval");       //Command Name
VariantInit(&exe_param);                                //Command Parameters
VariantInit(&exe_result);                               //Command Result

//Populate parameter option
exe_param.vt = VT_I4;
exe_param.intVal = 8;       //Logging interval to be set

/* SetCtrlLogInterval Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.113. CaoRobot::Execute("DetectOn ") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"DetectOn"); //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

//Populate parameter option
uint32_t *param_data;
```

```
exe_param.vt = VT_I4 | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_I4, 0, 5);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0] = 8;          //Input Signal port number to use as trigger
param_data[1] = 257;        //Data type to store
param_data[2] = 2;          //The smallest index number of the global variables to store
param_data[3] = 10;         //Number of data to be stored in the global variable
param_data[4] = 11;         //Index number of the integer type global variable to store
                            //the count of

//input signals which becomes triggers during the command enabled
SafeArrayUnaccessData(exe_param.parray);

/* DetectOn Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.114. CaoRobot::Execute("DetectOff ") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"DetectOff"); //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

//Populate parameter option
uint32_t *param_data;
exe_param.vt = VT_I4 | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_I4, 0, 2);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0] = 8;   //Port number to be disabled its function
param_data[1] = 0;   //Input signal detection edge
SafeArrayUnaccessData(exe_param.parray);

/* DetectOff Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.115. CaoRobot::Execute("GetPluralServoData ") command

Example

```
BSTR exe_comm;                                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"GetPluralServoData");       //Command Name
VariantInit(&exe_param);                                //Command Parameters
VariantInit(&exe_result);                               //Command Result

/* GetPluralServoData Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.28.116. CaoRobot::Execute("AngularTrigger ") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"AngularTrigger");   //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

//Populate parameter option
VARIANT *param_data;
exe_param.vt = VT_VARIANT | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_VARIANT, 0, 4);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0].intVal = 1;   //Valid/Invalid
aram_data[0].vt = VT_I4;
param_data[1].intVal = 1;   //Target Joint
param_data[1].vt = VT_I4;
param_data[2].intVal = 24;  //I/O Number to turn ON/OFF
param_data[2].vt = VT_I4;
param_data[3].dblVal = 10;  //Motion distance
param_data[3].vt = VT_R8;
SafeArrayUnaccessData(exe_param.parray);

/* AngularTrigger Command */
hr = bCap_RobotExecute(fd, hRobot, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_RobotExecute Succeeded...\n");
else
 printf("bCap_RobotExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.29. CaoTask::AddVariable method

5.2.30. CaoTask::get_VariableNames property

5.2.31. CaoTask::Start method

5.2.32. CaoTask::Stop method

5.2.33. CaoTask::Execute method

Example

```
//Get Task Handler
BSTR task_name, task_opt;                 //Arguments:
task_name = SysAllocString(L"Pro1");      //Name
task_opt = SysAllocString(L"");           //Option

/* Obtain Task Reference */
hr = bCap_ControllerGetTask(fd, hCtrl, task_name, task_opt, &hTask);

if (SUCCEEDED(hr))
 printf("bCap_ControllerGetTask Succeeded...\n");
else
 printf("bCap_ControllerGetTask Failed...\n");

BSTR exe_comm;                            //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"GetStatus"); //Command Name
VariantInit(&exe_param);                  //Command Parameters
VariantInit(&exe_result);                 //Command Result

/* Get Task Status */
hr = bCap_TaskExecute(fd, hTask, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_TaskExecute Succeeded... \n");
else
 printf("bCap_TaskExecute Failed... \n");

//Release Variables
SysFreeString(task_name);
SysFreeString(task_opt);
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.33.1. CaoTask::Execute("GetStatus") command

Example

```
BSTR exe_comm;                            //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"GetStatus"); //Command Name
VariantInit(&exe_param);                  //Command Parameters
VariantInit(&exe_result);                 //Command Result

/* Get Task Status */
hr = bCap_TaskExecute(fd, hTask, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_TaskExecute Succeeded... \n");
```

```
else
 printf("bCap_TaskExecute Failed... \n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.33.2. CaoTask::Execute("GetThreadPriority") command

5.2.33.3. CaoTask::Execute("SetThreadPriority") command

5.2.34. CaoVariable::get_Value property

5.2.35. CaoVariable::put_Value property

5.2.36. CaoExtension::Execute method

Example

```
//Get Extension Handler
BSTR ext_name, ext_opt;                    //Arguments:
ext_name = SysAllocString(L"Hand0");       //Name
ext_opt = SysAllocString(L"");             //Option

/* Obtain Extension Reference */
hr = bCap_ControllerGetExtension(fd, hCtrl, ext_name, ext_opt, &hExtension);
if (SUCCEEDED(hr))
 printf("bCap_ControllerGetExtension Succeeded...\n");
else
 printf("bCap_ControllerGetExtension Failed...\n");

//Release Variables
SysFreeString(ext_name);
SysFreeString(ext_opt);
```

5.2.36.1. Hand object - CaoExtension::Execute("Chuck") command

Example

```
BSTR exe_comm;                             //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"Chuck");       //Command Name
VariantInit(&exe_param);                   //Command Parameters
VariantInit(&exe_result);                  //Command Result

//Populate Parameter Option
exe_param.vt = VT_I4;
exe_param.intVal = 0;       //Point Number

/* Chuck Command */
hr = bCap_ExtensionExecute(fd, hExtension, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_ExtensionExecute Succeeded... \n");
else
 printf("bCap_ExtensionExecute Failed... \n");

//Release Variables
SysFreeString(exe_comm);
```

```
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.36.2. Hand object - CaoExtension::Execute("UnChuck") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"UnChuck");  //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

//Populate Parameter Option
exe_param.vt = VT_I4;
exe_param.intVal = 1;       //Point Number

/* UnChuck Command */
hr = bCap_ExtensionExecute(fd, hExtension, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_ExtensionExecute Succeeded... \n");
else
 printf("bCap_ExtensionExecute Failed... \n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.36.3. Hand object - CaoExtension::Execute("Motor") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"Motor");    //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

//Populate Parameter Option
exe_param.vt = VT_I4;
exe_param.intVal = 1;       //Motor Status

/* Motor Command */
hr = bCap_ExtensionExecute(fd, hExtension, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_ExtensionExecute Succeeded... \n");
else
 printf("bCap_ExtensionExecute Failed... \n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.36.4. Hand object - CaoExtension::Execute("Org") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"Org");      //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

/* Org Command */
hr = bCap_ExtensionExecute(fd, hExtension, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_ExtensionExecute Succeeded... \n");
else
 printf("bCap_ExtensionExecute Failed... \n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.36.5. Hand object - CaoExtension::Execute("MoveP") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"MoveP");    //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

//Populate Parameter Option
exe_param.vt = VT_I4;
exe_param.intVal = 1;       //Point Number

/* MoveP Command */
hr = bCap_ExtensionExecute(fd, hExtension, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_ExtensionExecute Succeeded... \n");
else
 printf("bCap_ExtensionExecute Failed... \n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.36.6. Hand object - CaoExtension::Execute("MoveA") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"MoveA");    //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

//Populate parameter option
VARIANT *param_data;
```

```
exe_param.vt = VT_VARIANT | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_VARIANT, 0, 2);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0].fltVal = 5.00;       //Position
param_data[0].vt = VT_R4;
param_data[1].intVal = 20;         //Speed
param_data[1].vt = VT_I4;
SafeArrayUnaccessData(exe_param.parray);

/* MoveA Command */
hr = bCap_ExtensionExecute(fd, hExtension, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_ExtensionExecute Succeeded...\n");
else
 printf("bCap_ExtensionExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.36.7. Hand object - CaoExtension::Execute("MoveR") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"MoveR");    //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

//Populate parameter option
VARIANT *param_data;
exe_param.vt = VT_VARIANT | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_VARIANT, 0, 2);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0].fltVal = -3.00;      //Position
param_data[0].vt = VT_R4;
param_data[1].intVal = 100;        //Speed
param_data[1].vt = VT_I4;
SafeArrayUnaccessData(exe_param.parray);

/* MoveR Command */
hr = bCap_ExtensionExecute(fd, hExtension, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_ExtensionExecute Succeeded...\n");
else
 printf("bCap_ExtensionExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.36.8. Hand object - CaoExtension::Execute("MoveAH") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"MoveAH");   //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

//Populate parameter option
VARIANT *param_data;
exe_param.vt = VT_VARIANT | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_VARIANT, 0, 3);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0].fltVal = 2.50;       //Position
param_data[0].vt = VT_R4;
param_data[1].intVal = 100;        //Speed
param_data[1].vt = VT_I4;
param_data[2].intVal = 100;        //Force
param_data[2].vt = VT_I4;
SafeArrayUnaccessData(exe_param.parray);

/* MoveAH Command */
hr = bCap_ExtensionExecute(fd, hExtension, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_ExtensionExecute Succeeded...\n");
else
 printf("bCap_ExtensionExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.36.9. Hand object - CaoExtension::Execute("MoveRH") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"MoveRH");   //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

//Populate parameter option
VARIANT *param_data;
exe_param.vt = VT_VARIANT | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_VARIANT, 0, 3);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0].fltVal = 2.50;       //Position
param_data[0].vt = VT_R4;
param_data[1].intVal = 100;        //Speed
param_data[1].vt = VT_I4;
param_data[2].intVal = 100;        //Force
param_data[2].vt = VT_I4;
SafeArrayUnaccessData(exe_param.parray);
```

```
/* MoveRH Command */
hr = bCap_ExtensionExecute(fd, hExtension, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_ExtensionExecute Succeeded...\n");
else
 printf("bCap_ExtensionExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.36.10. Hand object - CaoExtension::Execute("MoveH") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"MoveH");    //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

//Populate parameter option
uint32_t *param_data;
exe_param.vt = VT_I4 | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_I4, 0, 3);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0] = 50;         //Speed
param_data[1] = 100;        //Force
param_data[2] = 1;          //Direct
SafeArrayUnaccessData(exe_param.parray);

/* MoveH Command */
hr = bCap_ExtensionExecute(fd, hExtension, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_ExtensionExecute Succeeded...\n");
else
 printf("bCap_ExtensionExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.36.11 Hand object – CaoExtension::Execute(“MoveZH”) command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"MoveZH");   //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

//Populate parameter option
VARIANT *param_data;
exe_param.vt = VT_VARIANT | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_VARIANT, 0, 5);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
```

```
param_data[0].fltVal = 1.00;       //ZON1
param_data[0].vt = VT_R4;
param_data[1].fltVal = 4.00;       //ZON2
param_data[1].vt = VT_R4;
param_data[2].intVal = 50;         //Speed
param_data[2].vt = VT_I4;
param_data[3].intVal = 100;        //Force
param_data[3].vt = VT_I4;
param_data[4].intVal = 1;          //Direct
param_data[4].vt = VT_I4;
SafeArrayUnaccessData(exe_param.parray);

/* MoveZH Command */
hr = bCap_ExtensionExecute(fd, hExtension, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_ExtensionExecute Succeeded...\n");
else
 printf("bCap_ExtensionExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.36.12 Hand object – CaoExtension::Execute("Stop") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"Stop");     //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

/* Stop Command */
hr = bCap_ExtensionExecute(fd, hExtension, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_ExtensionExecute Succeeded... \n");
else
 printf("bCap_ExtensionExecute Failed... \n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.36.13 Hand object – CaoExtension::Execute("CurPos") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"CurPos");   //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

/* CurPos Command */
hr = bCap_ExtensionExecute(fd, hExtension, exe_comm, exe_param, &exe_result);
```

```
if (SUCCEEDED(hr))
 printf("bCap_ExtensionExecute Succeeded... \n");
else
 printf("bCap_ExtensionExecute Failed... \n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.36.14 Hand object – CaoExtension::Execute("GetPoint") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"GetPoint");  //Command Name
VariantInit(&exe_param);                 //Command Parameters
VariantInit(&exe_result);                //Command Result

//Populate parameter option
uint32_t *param_data;
exe_param.vt = VT_I4 | VT_ARRAY;
exe_param.parray = SafeArrayCreateVector(VT_I4, 0, 2);
hr = SafeArrayAccessData(exe_param.parray, (void**)&param_data);
param_data[0] = 0;   //Point Number
param_data[1] = 2;   //Point Data Element
SafeArrayUnaccessData(exe_param.parray);

/* GetPoint Command */
hr = bCap_ExtensionExecute(fd, hExtension, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_ExtensionExecute Succeeded...\n");
else
 printf("bCap_ExtensionExecute Failed...\n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.36.15 Hand object – CaoExtension::Execute("get_EmgState") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"get_EmgState");     //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

/* get_EmgState Command */
hr = bCap_ExtensionExecute(fd, hExtension, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_ExtensionExecute Succeeded... \n");
else
 printf("bCap_ExtensionExecute Failed... \n");
```

```
//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.36.16 Hand object – CaoExtension::Execute("get_ZonState") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"get_ZonState");     //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

/* get_ZonState Command */
hr = bCap_ExtensionExecute(fd, hExtension, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_ExtensionExecute Succeeded... \n");
else
 printf("bCap_ExtensionExecute Failed... \n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.36.17 Hand object – CaoExtension::Execute("get_OrgState") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"get_OrgState");     //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

/* get_OrgState Command */
hr = bCap_ExtensionExecute(fd, hExtension, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_ExtensionExecute Succeeded... \n");
else
 printf("bCap_ExtensionExecute Failed... \n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.36.18 Hand object – CaoExtension::Execute("get_HoldState") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"get_HoldState");    //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result
```

```
/* get_HoldState Command */
hr = bCap_ExtensionExecute(fd, hExtension, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_ExtensionExecute Succeeded... \n");
else
 printf("bCap_ExtensionExecute Failed... \n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.36.19 Hand object – CaoExtension::Execute("get_InposState") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"get_InposState");   //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

/* get_InposState Command */
hr = bCap_ExtensionExecute(fd, hExtension, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_ExtensionExecute Succeeded... \n");
else
 printf("bCap_ExtensionExecute Failed... \n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.36.20 Hand object – CaoExtension::Execute("get_Error") command

Example

```
BSTR exe_comm;                          //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"get_Error"); //Command Name
VariantInit(&exe_param);                //Command Parameters
VariantInit(&exe_result);               //Command Result

/* get_Error Command */
hr = bCap_ExtensionExecute(fd, hExtension, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_ExtensionExecute Succeeded... \n");
else
 printf("bCap_ExtensionExecute Failed... \n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.36.21 Hand object – CaoExtension::Execute("get_BusyState") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"get_BusyState");    //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

/* get_BusyState Command */
hr = bCap_ExtensionExecute(fd, hExtension, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_ExtensionExecute Succeeded... \n");
else
 printf("bCap_ExtensionExecute Failed... \n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```

5.2.36.22 Hand object – CaoExtension::Execute("get_MotorState") command

Example

```
BSTR exe_comm;                                  //Arguments:
VARIANT exe_param, exe_result;
exe_comm = SysAllocString(L"get_MotorState");   //Command Name
VariantInit(&exe_param);                        //Command Parameters
VariantInit(&exe_result);                       //Command Result

/* get_MotorState Command */
hr = bCap_ExtensionExecute(fd, hExtension, exe_comm, exe_param, &exe_result);
if (SUCCEEDED(hr))
 printf("bCap_ExtensionExecute Succeeded... \n");
else
 printf("bCap_ExtensionExecute Failed... \n");

//Release Variables
SysFreeString(exe_comm);
VariantClear(&exe_param);
VariantClear(&exe_result);
```