### 6.2.9 Communication port types

Ports facilitate communication between test components and between test components and the test system interface.

TTCN‑3 supports message-based and procedure-based ports. Each port shall be defined as being message-based or procedure-based. Message-based ports shall be identified by the keyword **message** and procedure-based ports shall be identified by the keyword **procedure** within the associated port type definition.

Ports are bidirectional. The directions are specified by the keywords **in** (for the in direction), **out** (for the out direction) and **inout** (for both directions). Directions shall be seen from the point of view of the test component owning the port with the exception of the test system interface, where **in** identifies the direction of message sending or procedure call and **out** identifies the direction of message receive, get reply or catch exception from the point of view of the test component connected to the test system interface port.

Each port type definition shall have one or more lists indicating the allowed collection of (message) types or procedure signatures together with the allowed communication direction.

For configuration purposes the port type may have one **map** **param** and one **unmap param** declaration indicating the allowed additional parameters for the respective operation. These formal parameters must be value parameters.

Whenever a signature (see also clause 14) is defined in the **out** direction of a procedure-based port, the types of all its **inout** and **out** parameters, its return type and its exception types are automatically part of the **in** direction of this port. Whenever a signature is defined in the **in** direction for a procedure-based port, the types of all its **inout** and **out** parameters, its return type and its exception types are automatically part of the **out** direction of this port.

Ports used for the communication with the SUT may need to address specific entities within the SUT. In addition, several address schemes may be supported by one SUT at different ports. To support such addressing schemes, TTCN-3 allows to bind an **address** type to a port. Values of this type may be used for addressing purposes in communication operations (see clause 22.1) and be stored in variables. The handling of address types bound to different ports by means of the dot notation is explained in clause 6.2.12.

***Syntactical Structure***

Message-based port:

**type** **port** *PortTypeIdentifier* **message** "{"

[ **address** *Type* “;” ]

 { (( **in** | **out** | **inout** ) { *MessageType* [ "," ] }+ |

 map param "(" { *FormalValuePar* [","] }+ ")" |

 unmap param "(" { *FormalValuePar* [","] }+ ")") ";" }

"}"

Procedure-based port:

**type** **port** *PortTypeIdentifier* **procedure** "{"

[ **address** *Type* “;” ]

 { (( **in** | **out** | **inout** ) { *Signature* [ "," ] }+ |

 map param "(" { *FormalValuePar* [","] }+ ")" |

 unmap param "(" { *FormalValuePar* [","] }+ ")") ";" }

"}"

***Restrictions***

No specific restrictions in addition to the general static rules of TTCN‑3 given in clause 5.

…

49. PortDefAttribs ::= MessageAttribs |

 ProcedureAttribs |

 MixedAttribs

50. MessageAttribs ::= MessageKeyword "{" [AddressDecl] {(MessageList | ConfigParamDef) [SemiColon]}+

 "}"

51. ConfigParamDef ::= MapParamDef | UnmapParamDef

52. MapParamDef ::= [MapKeyword ParamKeyword "(" FormalValuePar { "," FormalValuePar}")"]

53. UnmapParamDef ::= [UnmapKeyword ParamKeyword "(" FormalValuePar { "," FormalValuePar}")"]

54. AddressDecl ::= AddressKeyword Type SemiColon

55. MessageList ::= Direction AllOrTypeList

56. Direction ::= InParKeyword |

 OutParKeyword |

 InOutParKeyword

57. MessageKeyword ::= "message"

58. AllOrTypeList ::= AllKeyword | TypeList

/\* NOTE: The use of AllKeyword in port definitions is deprecated \*/

59. AllKeyword ::= "all"

60. TypeList ::= Type {"," Type}

61. ProcedureAttribs ::= ProcedureKeyword "{" [AddressDecl] {(ProcedureList | ConfigParamDef)

 [SemiColon]}+

 "}"

62. ProcedureKeyword ::= "procedure"

63. ProcedureList ::= Direction AllOrSignatureList

64. AllOrSignatureList ::= AllKeyword | SignatureList

65. SignatureList ::= Signature {"," Signature}

66. MixedAttribs ::= MixedKeyword "{" [AddressDecl] {(MixedList | ConfigParamDef) [SemiColon]}+

 "}"