8.6.1 Changes to TriCommunicationSA

The extension to the TriCommunicationSA class is mapped to the following class:

class xTriCommunicationSA {

public:

//Destructor.

virtual ~xTriCommunicationSA ();

//To establish a dynamic connection between two ports.

virtual TriStatus xtriMapParam (const TriPortId \*comPortId, const TriPortId \*tsiPortId,   
 TciParameterList \*parameterList)=0;

//To close a dynamic connection to the SUT for the referenced TSI port.

virtual TriStatus xtriUnmapParam (const TriPortId \*comPortId, const TriPortId \*tsiPortId,   
 TciParameterList \*parameterList)=0;

//Send operation on a component which has been mapped to a TSI port.

virtual TriStatus xtriSend (const TriComponentId \*componentId, const TriPortId \*tsiPortId, const TciValue \*SUTaddress, const TciValue \*sendMessage)=0;

//Send (broadcast) operation on a component which has been mapped to a TSI port.

virtual TriStatus xtriSendBC (const TriComponentId \*componentId, const TriPortId \*tsiPortId, const TciValue \*sendMessage)=0;

//Send (multicast) operation on a component which has been mapped to a TSI port.

virtual TriStatus xtriSendMC (const TriComponentId \*componentId, const TriPortId \*tsiPortId, const TciValueList \*SUTaddresses, const TciValue \*sendMessage)=0;

//Initiate the procedure call.

virtual TriStatus xtriCall (const TriComponentId \*componentId, const TriPortId \*tsiPortId, const TciValue \*sutAddress, const TriSignatureId \*signatureId, const TciParameterList \*parameterList)=0;

//Initiate and broadcast the procedure call.

virtual TriStatus xtriCallBC (const TriComponentId \*componentId, const TriPortId \*tsiPortId, const TriSignatureId \*signatureId, const TciParameterList \*parameterList)=0;

//Initiate and multicast the procedure call.

virtual TriStatus xtriCallMC (const TriComponentId \*componentId, const TriPortId \*tsiPortId, const TciValueList \*sutAddresses, const TriSignatureId \*signatureId, const TciParameterList \*parameterList)=0;

//Issue the reply to a procedure call.

virtual TriStatus xtriReply (const TriComponentId \*componentId, const TriPortId \*tsiPortId, const TciValue \*sutAddress, const TriSignatureId \*signatureId, const TciParameterList \* parameterList, const TciValue \*returnValue)=0;

//Broadcast the reply to a procedure call.

virtual TriStatus xtriReplyBC (const TriComponentId \*componentId, const TriPortId \*tsiPortId, const TriSignatureId \*signatureId, const TciParameterList \*parameterList, const TciValue \*returnValue)=0;

//Multicast the reply to a procedure call.

virtual TriStatus xtriReplyMC (const TriComponentId \*componentId, const TriPortId \*tsiPortId, const TciValueList \*sutAddresses, const TriSignatureId \*signatureId, const TciParameterList \*parameterList, const TciValue \*returnValue)=0;

//Raise an exception to a procedure call.

virtual TriStatus xtriRaise (const TriComponentId \*componentId, const TriPortId \*tsiPortId, const TciValue \*sutAddress, const TriSignatureId \*signatureId, const TciValue \*exc)=0;

//Raise a broadcast an exception to a procedure call.

virtual TriStatus xtriRaiseBC (const TriComponentId \*componentId, const TriPortId \*tsiPortId, const TriSignatureId \*signatureId, const TciValue \*exc)=0;

//Raise a multicast an exception to a procedure call.

virtual TriStatus xtriRaiseMC (const TriComponentId \*componentId, const TriPortId \*tsiPortId, const TciValueList \*sutAddresses, const TriSignatureId \*signatureId, const TciValue \*exc)=0;

// Miscellaneous operations

virtual TciValue \*xtriConvert(const Object \*value, const TciType \*typeHypothesis)=0;

}

8.6.2 Changes to TriCommunicationTE

The extension to the TriCommunicationTE class is mapped to the following class:

class xTriCommunicationTE {

public:

//Destructor.

virtual ~xTriCommunicationTE ();

//Called by SA after it has received a message from the SUT.

virtual void xtriEnqueueMsg (const TriPortId \*tsiPortId, const Object \*SUTaddress, const TriComponentId \*componentId, const Object \*receivedMessage)=0;

//Called by SA after it has received a procedure call from the SUT.

virtual void xtriEnqueueCall (const TriPortId \*tsiPortId, const Object \*SUTaddress, const TriComponentId \*componentId, const TriSignatureId \*signatureId, const TciParameterList \*parameterList)=0;

//Called by SA after it has received a reply from the SUT.

virtual void xtriEnqueueReply (const TriPortId \*tsiPortId, const Object \*SUTaddress, const TriComponentId \*componentId, const TriSignatureId \*signatureId, const TciParameterList \*parameterList, const TciValue \*returnValue)=0;

//Called by SA after it has received an exception from the SUT.

virtual void xtriEnqueueException (const TriPortId \*tsiPortId, const Object \*SUTaddress, const TriComponentId \*componentId, const TriSignatureId \*signatureId, const Object \*exc)=0;

// Error handling

virtual void xtriSAErrorReq (const String message, const Object \*cause)=0;

}

8.6.3 Changes to TriPlatformPA

The extension to the TriPlatformPA class is mapped to the following class:

class xTriPlatformPA {

public:

//Destructor.

virtual ~xTriPlatformPA ();

//For each external function specified in the TTCN-3 ATS implement the behaviour.

virtual TriStatus xtriExternalFunction (const TriFunctionId \*functionId, TciParameterList \*parameterList, TciValue \*returnValue)=0;

}

8.6.4 Changes to TriPlatformTE

The extension to the TriPlatformTE class is mapped to the following interface:

class xTriPlatformTE {

public:

//Destructor.

virtual ~xTriPlatformTE ();

//Called by PA in unrecoverable error situations.

virtual void xtriPAError (const Tstring &message, const Object \*cause)=0;

//Generate random number.

virtual FloatValue\* xtriRnd (const TriComponentId \*componentId, const FloatValue \*seed)=0;

}