### 12.5.3 TCI-CH interface

#### 12.5.3.1 TCI-CH provided

The **TCI-CH provided** interface is mapped to the following interface:

public interface ITciCHProvided {
 void TciSendConnected(Etsi.Ttcn3.Tri.ITriPortId sender,
 Etsi.Ttcn3.Tri.ITriComponentId receiver, ITciValue sendMessage);
 void TciSendConnectedBC(Etsi.Ttcn3.Tri.ITriPortId sender,
 ITciValue sendMessage);
 void TciSendConnectedMC(Etsi.Ttcn3.Tri.ITriPortId sender,
 Etsi.Ttcn3.Tri.ITriComponentIdList receivers,
 ITciValue sendMessage);
 void TciCallConnected(Etsi.Ttcn3.Tri.ITriPortId sender,
 Etsi.Ttcn3.Tri.ITriComponentId receiver,
 Etsi.Ttcn3.Tri.ITriSignatureId signature,
 ITciParameterList parameterList);
 void TciCallConnectedBC(Etsi.Ttcn3.Tri.ITriPortId sender,
 Etsi.Ttcn3.Tri.ITriSignatureId signature,
 ITciParameterList parameterList);
 void TciCallConnectedMC(Etsi.Ttcn3.Tri.ITriPortId sender,
 Etsi.Ttcn3.Tri.ITriComponentIdList receivers,
 Etsi.Ttcn3.Tri.ITriSignatureId signature,
 ITciParameterList parameterList);
 void TciReplyConnected(Etsi.Ttcn3.Tri.ITriPortId sender,
 Etsi.Ttcn3.Tri.ITriComponentId receiver,
 Etsi.Ttcn3.Tri.ITriSignatureId signature,
 ITciParameterList parameterList, ITciValue returnValue);
 void TciReplyConnectedBC(Etsi.Ttcn3.Tri.ITriPortId sender,
 Etsi.Ttcn3.Tri.ITriSignatureId signature,
 ITciParameterList parameterList, ITciValue returnValue);
 void TciReplyConnectedMC(Etsi.Ttcn3.Tri.ITriPortId sender,
 Etsi.Ttcn3.Tri.ITriComponentIdList receivers,
 Etsi.Ttcn3.Tri.ITriSignatureId signature,
 ITciParameterList parameterList, ITciValue returnValue);
 void TciRaiseConnected(Etsi.Ttcn3.Tri.ITriPortId sender,
 Etsi.Ttcn3.Tri.ITriComponentId receiver,
 Etsi.Ttcn3.Tri.ITriSignatureId signature, ITciValue except);
 void TciRaiseConnectedBC(Etsi.Ttcn3.Tri.ITriPortId sender,
 Etsi.Ttcn3.Tri.ITriSignatureId signature, ITciValue except);
 void TciRaiseConnectedMC(Etsi.Ttcn3.Tri.ITriPortId sender,
 Etsi.Ttcn3.Tri.ITriComponentIdList receivers,
 Etsi.Ttcn3.Tri.ITriSignatureId signature, ITciValue except);
 Etsi.Ttcn3.Tri.ITriComponentId TciCreateTestComponentReq(int kind,
 ITciType componentType, string name, ITciValue hostId);
 void TciStartTestComponentReq(Etsi.Ttcn3.Tri.ITriComponentId comp,
 ITciBehaviourId behavior, ITciParameterList parameterList);
 void TciStopTestComponentReq(Etsi.Ttcn3.Tri.ITriComponentId comp);
 void TciConnectReq(Etsi.Ttcn3.Tri.ITriPortId fromPort,
 Etsi.Ttcn3.Tri.ITriPortId toPort);
 void TciDisconnectReq(Etsi.Ttcn3.Tri.ITriPortId fromPort,
 Etsi.Ttcn3.Tri.ITriPortId toPort);
 void TciTestComponentTerminatedReq(Etsi.Ttcn3.Tri.ITriComponentId comp,
 ITciVerdictValue verdict);
 bool TciTestComponentRunningReq(Etsi.Ttcn3.Tri.ITriComponentId comp);
 Etsi.Ttcn3.Tri.ITriComponentId TciGetMmcReq();
 void TciMapReq(Etsi.Ttcn3.Tri.ITriPortId fromPort,
 Etsi.Ttcn3.Tri.ITriPortId toPort);

 void TciMapParamReq(Etsi.Ttcn3.Tri.ITriPortId fromPort,
 Etsi.Ttcn3.Tri.ITriPortId toPort, Etsi.Ttcn3.Tci.ITciParameterList parameterList);
 void TciUnmapReq(Etsi.Ttcn3.Tri.ITriPortId fromPort,
 Etsi.Ttcn3.Tri.ITriPortId toPort);

 void TciUnmapParamReq(Etsi.Ttcn3.Tri.ITriPortId fromPort,
 Etsi.Ttcn3.Tri.ITriPortId toPort, Etsi.Ttcn3.Tci.ITciParameterList parameterList);
 void TciExecuteTestCaseReq(Etsi.Ttcn3.Tri.ITriComponentId component,
 Etsi.Ttcn3.Tri.ITriPortIdList tsiPortList);
 void TciResetReq();
 bool TciTestComponentDoneReq(Etsi.Ttcn3.Tri.ITriComponentId component);
 void TciKillTestComponentReq(Etsi.Ttcn3.Tri.ITriComponentId component);
 bool TciTestComponentAliveReq(Etsi.Ttcn3.Tri.ITriComponentId component);
 bool TciTestComponentKilledReq(Etsi.Ttcn3.Tri.ITriComponentId component);
}

#### 12.5.3.2 TCI-CH required

The **TCI-CH required** interface is mapped to the following interface:

public interface ITciCHRequired {
 void TciEnqueueMsgConnected(Etsi.Ttcn3.Tri.ITriPortId sender,
 Etsi.Ttcn3.Tri.ITriComponentId receiver,
 ITciValue receivedMessage);
 void TciEnqueueCallConnected(Etsi.Ttcn3.Tri.ITriPortId sender,
 Etsi.Ttcn3.Tri.ITriComponentId receiver,
 Etsi.Ttcn3.Tri.ITriSignatureId signature,
 ITciParameterList parameterList);
 void TciEnqueueReplyConnected(Etsi.Ttcn3.Tri.ITriPortId sender,
 Etsi.Ttcn3.Tri.ITriComponentId receiver,
 Etsi.Ttcn3.Tri.ITriSignatureId signature,
 ITciParameterList parameterList, ITciValue returnValue);
 void TciEnqueueRaiseConnected(Etsi.Ttcn3.Tri.ITriPortId sender,
 Etsi.Ttcn3.Tri.ITriComponentId receiver,
 Etsi.Ttcn3.Tri.ITriSignatureId signature, ITciValue except);
 Etsi.Ttcn3.Tri.ITriComponentId TciCreateTestComponent(int kind,
 ITciType componentType, string name);
 void TciStartTestComponent(Etsi.Ttcn3.Tri.ITriComponentId comp,
 ITciBehaviourId behavior, ITciParameterList parameterList);
 void TciStopTestComponent(Etsi.Ttcn3.Tri.ITriComponentId comp);
 void TciConnect(Etsi.Ttcn3.Tri.ITriPortId fromPort,
 Etsi.Ttcn3.Tri.ITriPortId toPort);
 void TciDisconnect(Etsi.Ttcn3.Tri.ITriPortId fromPort,
 Etsi.Ttcn3.Tri.ITriPortId toPort);
 void TciTestComponentTerminated(Etsi.Ttcn3.Tri.ITriComponentId comp,
 ITciVerdictValue verdict);
 bool TciTestComponentRunning(Etsi.Ttcn3.Tri.ITriComponentId comp);
 bool TciTestComponentDone(Etsi.Ttcn3.Tri.ITriComponentId comp);
 Etsi.Ttcn3.Tri.ITriComponentId TciGetMtc();
 void TciExecuteTestCase (ITciTestCaseId testCaseId,
 Etsi.Ttcn3.Tri.ITriPortIdList tsiPortList);
 void TciReset();
 void TciMap(Etsi.Ttcn3.Tri.ITriPortId fromPort,
 Etsi.Ttcn3.Tri.ITriPortId toPort);

 void TciMapParam(Etsi.Ttcn3.Tri.ITriPortId fromPort,
 Etsi.Ttcn3.Tri.ITriPortId toPort, Etsi.Ttcn3.Tci.ITciParameterList parameterList);
 void TciUnmap(Etsi.Ttcn3.Tri.ITriPortId fromPort,
 Etsi.Ttcn3.Tri.ITriPortId toPort);

 void TciUnmapParam(Etsi.Ttcn3.Tri.ITriPortId fromPort,
 Etsi.Ttcn3.Tri.ITriPortId toPort, Etsi.Ttcn3.Tci.ITciParameterList parameterList);
 void TciKillTestComponent(Etsi.Ttcn3.Tri.ITriComponentId component);
 bool TciTestComponentAlive (Etsi.Ttcn3.Tri.ITriComponentId component);
 bool TciTestComponentKilled(Etsi.Ttcn3.Tri.ITriComponentId component);
}