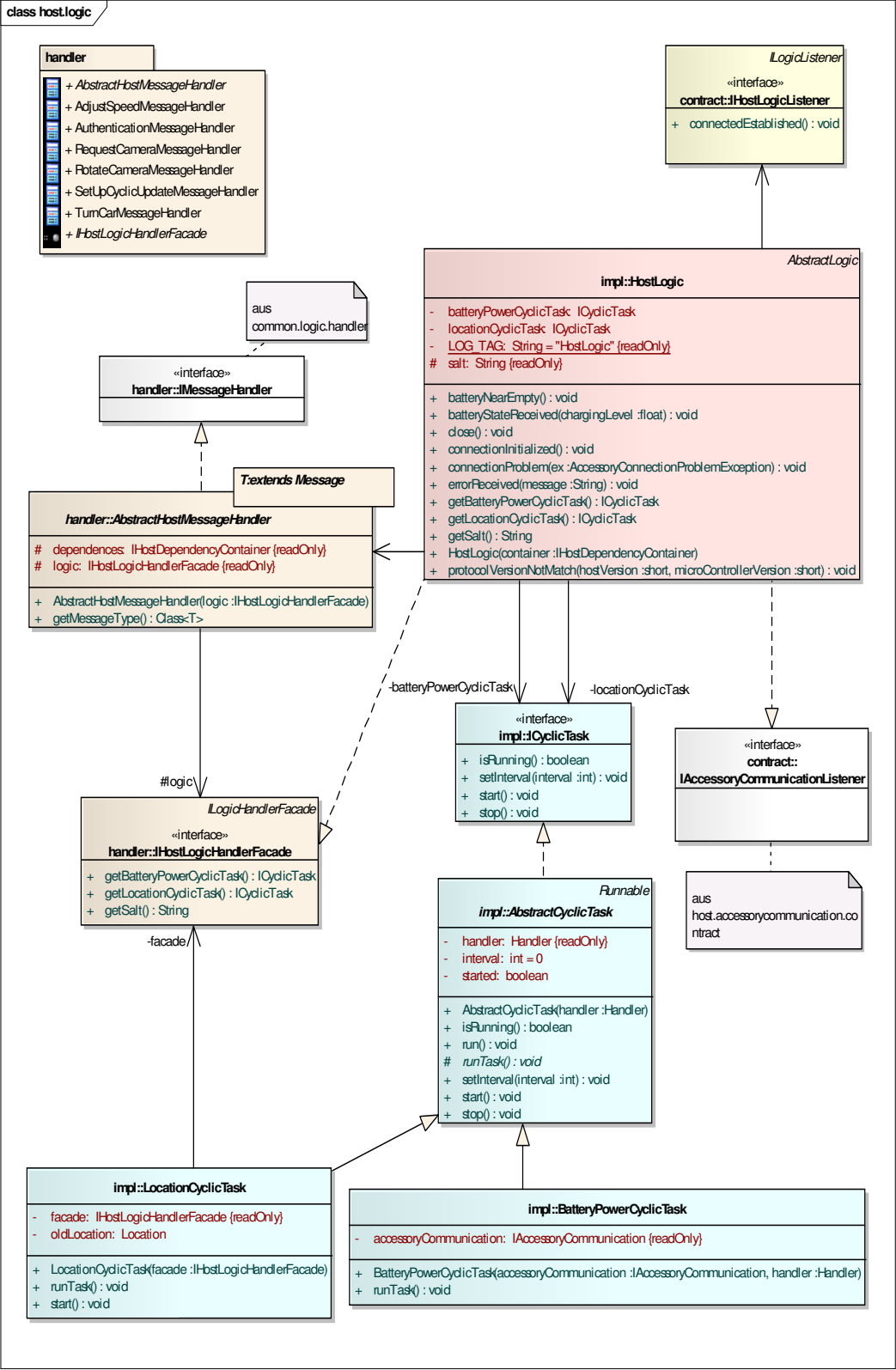
class host.logic
handler
+ AbstractHostMessageHandler
+ AdjustSpeedMessageHandler
+ AuthenticationMessageHandler
+ RequestCameraMessageHandler
+ RotateCameraMessageHandler
+ SetUpCyclicUpdateMessageHandler
+ TurnCarMessageHandler
+ IHostLogicHandlerFacade
ILogicListener
«interface»
contract::IHostLogicListener
+ connectedEstablished() : void
AbstractLogic
impl::HostLogic
- batteryPowerCyclicTask: ICyclicTask
- locationCyclicTask: ICyclicTask
- LOG_TAG: String = "HostLogic" {readOnly}
salt: String {readOnly}
+ batteryNearEmpty() : void
+ batteryStateReceived(chargingLevel :float) : void
+ close() : void
+ connectionInitialized() : void
+ connectionProblem(ex :AccessoryConnectionProblemException) : void
+ errorReceived(message :String) : void
+ getBatteryPowerCyclicTask() : ICyclicTask
+ getLocationCyclicTask() : ICyclicTask
+ getSalt() : String
+ HostLogic(container :IHostDependencyContainer)
+ protocolVersionNotMatch(hostVersion :short, microControllerVersion :short) : void
aus
common.logic.handler
«interface»
handler::IMessageHandler
T:extends Message
handler::AbstractHostMessageHandler
dependences: IHostDependencyContainer {readOnly}
logic: IHostLogicHandlerFacade {readOnly}
+ AbstractHostMessageHandler(logic :IHostLogicHandlerFacade)
+ getMessageType() : Class<T>
-batteryPowerCyclicTask
-locationCyclicTask
«interface»
impl::ICyclicTask
+ isRunning() : boolean
+ setInterval(interval :int) : void
+ start() : void
+ stop() : void
«interface»
contract::
IAccessoryCommunicationListener
#logic
ILogicHandlerFacade
«interface»
handler::IHostLogicHandlerFacade
+ getBatteryPowerCyclicTask() : ICyclicTask
+ getLocationCyclicTask() : ICyclicTask
+ getSalt() : String
-facade
Runnable
impl::AbstractCyclicTask
- handler: Handler {readOnly}
- interval: int = 0
- started: boolean
+ AbstractCyclicTask(handler :Handler)
+ isRunning() : boolean
+ run() : void
# runTask() : void
+ setInterval(interval :int) : void
+ start() : void
+ stop() : void
aus
host.accessorycommunication.contract
impl::LocationCyclicTask
- facade: IHostLogicHandlerFacade {readOnly}
- oldLocation: Location
+ LocationCyclicTask(facade :IHostLogicHandlerFacade)
+ runTask() : void
+ start() : void
impl::BatteryPowerCyclicTask
- accessoryCommunication: IAccessoryCommunication {readOnly}
+ BatteryPowerCyclicTask(accessoryCommunication :IAccessoryCommunication, handler :Handler)
+ runTask() : void