Testkonzept für eCAL – Integrationstestszenarien

April 28, 2025

Functional Tests

1. Basic Communication Tests

- **1.1 Publisher to Subscriber Communication**
  Ziel: Sicherstellen, dass eine Nachricht vom Publisher korrekt beim Subscriber ankommt.
  Prüfung: Empfang der Nachricht, Inhalt korrekt, keine Fehlermeldung.
- **1.2 Multiple Subscribers for One Publisher**
  Ziel: Testen, ob ein Publisher gleichzeitig mehrere Subscriber bedienen kann.
  Prüfung: Alle Subscriber empfangen dieselbe Nachricht.
- **1.3 Multiple Publishers on the Same Topic**
  Ziel: Überprüfen, wie eCAL mit mehreren Publishern auf demselben Topic umgeht.
  Prüfung: Subscriber empfängt korrekt von beiden (falls erlaubt), oder erkennt Konflikte.

2. Message Content and Format Validation

- **2.1 Correctness of Payload Data**
  Ziel: Sicherstellen, dass strukturierte Daten (z. B. Protobuf-Messages) korrekt übertragen werden.
  Prüfung: Felderinhalt auf Sender- und Empfängerseite identisch.
- **2.2 Handling of Malformed Messages**
  Ziel: Überprüfen, wie das System auf beschädigte oder unvollständige Nachrichten reagiert.
  Prüfung: Fehlerbehandlung, kein Crash, Logmeldung erzeugt.

4. Fault Injection and Recovery

- **4.1 Publisher and Subscriber Crash During Transmission**
  Ziel: Verhalten bei Absturz während laufender Kommunikation testen.
  Prüfung: Automatischer Reconnect oder Fehlererkennung, keine Systeminstabilität.
- **4.2 Network Crash Injection**
  Ziel: Simulieren eines Netzwerkausfalls (z. B. Interface deaktivieren).
  Prüfung: Fehlererkennung, ggf. Reconnect, Logging.

- **4.3 Packet Loss Simulation**
  Ziel: Verlust einzelner Pakete testen (z. B. durch tc/netem).
  Prüfung: Verhalten des Systems bei unvollständigen Nachrichten.
- **4.4 Node Reconnection**
  Ziel: Testen, ob ein Subscriber nach Restart korrekt erneut Verbindungen aufbaut.
  Prüfung: Empfang nach Reconnect funktioniert.
- **4.5 Empty Payloads**
  Ziel: Verhalten bei leeren Nachrichten untersuchen.
  Prüfung: Kein Crash, Logging korrekt, ggf. Fehlererkennung.

5. Scalability & Load

- **5.1 Start 100+ Subscriber Nodes**
  Ziel: Skalierbarkeit der Kommunikation bei hoher Subscriber-Anzahl testen.
  Prüfung: Alle Subscriber empfangen Nachrichten, keine Verzögerung oder Instabilität.
- **5.2 Multiple Topics Parallel**
  Ziel: Parallelverarbeitung mehrerer Topics durch Publisher/Subscriber testen.
  Prüfung: Nachrichtenfluss bleibt stabil, keine Interferenzen.

6. Service (RPC) Communication

- **6.1 Basic RPC Call**
  Ziel: Grundfunktionalität von RPC testen (Request/Response).
  Prüfung: Antwort korrekt empfangen, Roundtrip-Zeit messen.
- **6.2 Timeout Behavior for RPC**
  Ziel: Verhalten bei nicht beantworteten RPCs prüfen.
  Prüfung: Timeout erfolgt korrekt, Fehlerbehandlung vorhanden.

7. Topic Discovery and Registration

- **7.1 Automatic Topic Discovery**
  Ziel: Sicherstellen, dass neue Publisher automatisch vom System erkannt werden, ohne manuelle Konfiguration.
  Prüfung: Subscriber erkennt das neue Topic, sobald der Publisher aktiv wird, und beginnt korrekt zu empfangen.

8. Message Filtering

- **8.1 Subscriber Filters by Topic Name or Content**
  Ziel: Sicherstellen, dass ein Subscriber gezielt nur bestimmte Nachrichten verarbeitet (z. B. basierend auf Topic-Namen oder Nachrichteninhalt).
  Prüfung: Nur relevante Nachrichten werden empfangen, irrelevante ignoriert.
- **8.2 Priority-Based Processing (optional)**
  Ziel: Überprüfen, ob Nachrichten mit höherer Priorität bevorzugt verarbeitet werden.
  Prüfung: Verarbeitung erfolgt in der erwarteten Reihenfolge (Priorität $>$ Zeit).

Non-Functional Tests

9. Environment Tests

- **9.1 Cross-OS Testing (e.g. Linux ↔ Windows)**
  Ziel: Sicherstellen, dass Kommunikation plattformübergreifend funktioniert.
  Prüfung: Nachrichtenempfang und -versand unter gemischten Betriebssystemen.

10. Transport Layer Tests

- **10.1 Fallback to Alternative Transport**
  Ziel: Verhalten bei Ausfall eines Transports prüfen (z. B. UDP blockiert → TCP nutzen). UDP vs. TCP vs. Shared Memory
  Prüfung: Automatischer Wechsel, Stabilität erhalten, Fehlermeldung vorhanden.
- **10.3 Shared Memory Isolation Test**
  Ziel: Prüfen, ob zwei Prozesse in getrennten Namespaces (z. B. Docker-Containern) per Shared Memory kommunizieren können.
  Prüfung: Kommunikation funktioniert nur bei gleichem Host, ggf. Sicherheitsaspekt.

11. Network Topology Tests

- **11.1 Cross-Subnet Communication (optional)**
  Ziel: Prüfen, ob eCAL über Subnetzgrenzen hinweg kommuniziert.
  Prüfung: Sender/Empfänger in unterschiedlichen Docker-Netzen → manuell bridgen → Kommunikation testen.
- **11.2 Multicast Blocking Simulation (optional)**
  Ziel: Blockieren von Multicast-UDP (z. B. durch iptables) → Verhalten testen.
  Prüfung: Fallback, Fehlerbehandlung, Warnung.

12. Firewall & Port Configuration

- **12.1 Ports Blocked by Firewall (optional)**
  Ziel: Verhalten bei gesperrten Ports simulieren.
  Prüfung: Fehler erkannt, Alternativen genutzt.
- **12.2 Port Conflicts (optional)**
  Ziel: Zwei Prozesse belegen denselben Port.
  Prüfung: Korrekte Fehlererkennung, Logging.