

DIN EN 4660-005

ICS 49.090

**Luft- und Raumfahrt –
Modulare und offene Avionikarchitekturen –
Teil 005: Software;
Deutsche und Englische Fassung EN 4660-005:2011**

Aerospace series –
Modular and Open Avionics Architectures –
Part 005: Software;
German and English version EN 4660-005:2011

Série aérospatiale –
Architectures Avioniques Modulaires et Ouvertes –
Partie 005: Software;
Version allemande et anglaise EN 4660-005:2011

Gesamtumfang 955 Seiten

Normenausschuss Luft- und Raumfahrt (NL) im DIN

Nationales Vorwort

Der Verband der Europäischen Luft-, Raumfahrt- und Verteidigungsindustrie Normung (ASD-STAN) ist vom Europäischen Komitee für Normung (CEN) für zuständig erklärt worden, Europäische Normen (EN) für das Gebiet der Luft- und Raumfahrt auszuarbeiten. Durch die Vereinbarung vom 3. Oktober 1986 wurde ASD Assoziierte Organisation (ASB) des CEN.

Die vorliegende Norm EN 4660-005:2011 wurde von ASD-STAN, Fachbereich Allgemeines, unter Mitwirkung deutscher Experten des Normenausschusses Luft- und Raumfahrt erarbeitet.

Das zuständige deutsche Normungsgremium ist der Arbeitsausschuss NA 131-04-09 AA „Avionik“ im DIN-Normenausschuss Luft- und Raumfahrt.

Entsprechend Beschluss 57/9 des Technischen Ausschusses des Beirats des Normenausschusses Luft- und Raumfahrt (NL) im DIN Deutsches Institut für Normung e. V. sind die europäischen Luft- und Raumfahrt-Normungsergebnisse zweisprachig, in Deutsch und Englisch, in das Deutsche Normenwerk zu überführen. Aus diesem Grund wurde der Deutschen Fassung dieses Dokuments die Englische Fassung hinzugefügt.

Deutsche Fassung

Luft- und Raumfahrt —
Modulare und offene Avionikarchitekturen —
Teil 005: Software

Aerospace series —
Modular and Open Avionics Architectures —
Part 005: Software

Série aérospatiale —
Architectures Avioniques Modulaires et Ouvertes —
Partie 005: Software

Diese Europäische Norm wurde vom CEN am 26. Juni 2010 angenommen.

Die CEN-Mitglieder sind gehalten, die CEN/CENELEC-Geschäftsordnung zu erfüllen, in der die Bedingungen festgelegt sind, unter denen dieser Europäischen Norm ohne jede Änderung der Status einer nationalen Norm zu geben ist. Auf dem letzten Stand befindliche Listen dieser nationalen Normen mit ihren bibliographischen Angaben sind beim Management-Zentrum des CEN-CENELEC oder bei jedem CEN-Mitglied auf Anfrage erhältlich.

Diese Europäische Norm besteht in drei offiziellen Fassungen (Deutsch, Englisch, Französisch). Eine Fassung in einer anderen Sprache, die von einem CEN-Mitglied in eigener Verantwortung durch Übersetzung in seine Landessprache gemacht und dem Management-Zentrum mitgeteilt worden ist, hat den gleichen Status wie die offiziellen Fassungen.

CEN-Mitglieder sind die nationalen Normungsinstitute von Belgien, Bulgarien, Dänemark, Deutschland, Estland, Finnland, Frankreich, Griechenland, Irland, Island, Italien, Kroatien, Lettland, Litauen, Luxemburg, Malta, den Niederlanden, Norwegen, Österreich, Polen, Portugal, Rumänien, Schweden, der Schweiz, der Slowakei, Slowenien, Spanien, der Tschechischen Republik, Ungarn, dem Vereinigten Königreich und Zypern.



EUROPÄISCHES KOMITEE FÜR NORMUNG
EUROPEAN COMMITTEE FOR STANDARDIZATION
COMITÉ EUROPÉEN DE NORMALISATION

Management-Zentrum: Avenue Marnix 17, B-1000 Brüssel

Inhalt

	Seite
Vorwort	10
0 Einleitung.....	11
0.1 Zweck	11
0.2 Aufbau des Dokuments.....	12
1 Anwendungsbereich	12
1.1 Überblick über die Software-Architektur	12
1.2 Komponenten der Software-Architektur	13
2 Normative Verweisungen	15
3 Begriffe und Abkürzungen.....	15
3.1 Begriffe	16
3.2 Abkürzungen	16
4 Systemfunktionen.....	18
4.1 Systemverwaltungsfunktion.....	18
4.2 Kommunikation	25
4.3 Sicherheitsmanagement	41
4.4 Modulverwaltung	44
4.5 Massenspeicherverwaltung.....	45
4.6 Graphikverwaltung	49
4.7 Energieverwaltung.....	51
4.8 Netzwerkverwaltung	53
4.9 Zeitmanagement	56
5 Definition der Software-Architektur	59
5.1 MSL	61
5.2 OSL.....	65
5.3 RTBP	99
5.4 Anwendungsschicht.....	101
6 Definition direkter Schnittstellen	107
6.1 APOS	107
6.2 MOS	171
6.3 SMBP.....	245
6.4 SMOS	266
7 Definition logischer Schnittstellen.....	307
7.1 OLI	307
7.2 GLI	313
7.3 SMLI	334
7.4 MLI	342
8 Datentypdefinitionen	391
8.1 IDL	391
8.2 Datentypen	393
9 Implementierungsspezifischer Zuschnitt.....	434
Anhang A (normativ) AGL	443
A.1 Konzept.....	443
A.2 Grafikbefehlssatz	443
A.2.1 Überblick.....	443
A.2.2 Auflistung der Befehle	444
A.2.3 Definition der Hilfsbibliothek (Auxiliary Library, AL)	447
A.2.4 Definition der Videothek (Video Library, VL)	448
A.2.5 Grenzbedingungen für Texture Mapping	449
A.2.6 Einzelbild und Synchronisierung.....	451
A.2.7 Befehlsantworten und -verzögerungen	451

Bilder

Bild 1 — Hierarchie der Dokumentation der ASAAC-Standards	11
Bild 2 — Dreischichtige ASAAC-Software-Architektur	12
Bild 3 — Das Software-Architekturmodell	13
Bild 4 — Hierarchische Organisation der Systemverwaltung	19
Bild 5 — GSM-Aufgliederung für RE-Verwaltung (Beispiel)	20
Bild 6 — IA-Anwendungssteuerung (Beispiel)	21
Bild 7 — GSM-Aufgliederung für Modulverwaltung (Beispiel)	21
Bild 8 — Hierarchische Organisation des AM (Beispiel)	22
Bild 9 — Der ASAAC-Kommunikationsstapel	25
Bild 10 — Datentransferarten	27
Bild 11 — Kommunikationskonzept	28
Bild 12 — Routing der Kommunikation zwischen AL	28
Bild 13 — ASAAC-Nachricht bei BMC-Datentransfer	30
Bild 14 — Muticast-Schema mit einer einzelnen TC	31
Bild 15 — Muticast-Schema mit mehreren einfachen TCs	32
Bild 16 — Datenparallelität	33
Bild 17 — Corner-Turn	33
Bild 18 — Corner-Turn in drei Dimensionen	34
Bild 19 — Darstellung der beteiligten Dienste in DSP1	35
Bild 20 — Datendarstellung	37
Bild 21 — GSM-Schnittstellen	41
Bild 22 — Wesentliche sicherheitsbezogene Architekturkomponenten	42
Bild 23 — VC überträgt Daten, die einer Ver- und Entschlüsselung bedürfen	43
Bild 24 — Dateihandhabung durch eine entfernte Anwendung	46
Bild 25 — MMM-Download auf ein CFM, das nur die MSL besitzt	47
Bild 26 — CFM-Download auf ein CFM, das nur die MSL aufweist	48
Bild 27 — Anwendungsdownload	49
Bild 28 — Graphikkonzept	50
Bild 29 — Graphikstandard	50
Bild 30 — Anwendungssteuerung	52
Bild 31 — PCM-Verwaltung (Beispiel)	52

	Seite
Bild 32 — Konfiguration eines NSM.....	55
Bild 33 — Format der Netzwerkkonfigurationsnachricht	55
Bild 34 — Uhrenhierarchie in einem ASAAC-System	58
Bild 35 — Software-Architektur	60
Bild 36 — ASAAC-Softwarestapel auf einem CFM	61
Bild 37 — Logische GSM-Schnittstelle.....	66
Bild 38 — GSM: Externe Schnittstellen	67
Bild 39 — Diagramm des Zustandsübergangs von Threads mit Beispielen für APOS-Dienste.....	74
Bild 40 — Prozesszustandsdiagramm.....	78
Bild 41 — Beispiel für 1-zu-N-FIFO	81
Bild 42 — Beispiel für 1-zu-N-LIFO	82
Bild 43 — Verarbeitung eines Anwendungsfehlers durch das OS.....	89
Bild 44 — Verarbeitung eines MSL-Fehlers durch das OS aufgrund der Rückmeldung eines MOS-Dienstes	90
Bild 45 — Verarbeitung eines MSL-Fehlers durch das OS aufgrund eines CBIT-Status	91
Bild 46 — Die OLI	93
Bild 47 — Aufgliederung für OLI	93
Bild 48 — RTBP-Konzept im Baumdiagramm.....	99
Bild 49 — Zusammenhang zwischen Prozessen, Threads und VC	102
Bild 50 — Software-Architekturmodell – Dreischichtenstapel (TLS)	171
Bild 51 — MOS-Software-Architekturmodell.....	172
Bild 52 — Beschreibung des Datenpuffers von „sendFragmentedTransfer“	224
Bild 53 — Aufteilung eingehender Daten mit „receiveFragmentedTransfer“	227
Bild 54 — Unterschiedliche Schrittgrößen bei fragmentierter Übertragung	227
Bild 55 — Wurzeldefinition	248
Bild 56 — Funktionsmengendefinition	249
Bild 57 — Konfigurationsmengendefinition.....	250
Bild 58 — Prozessmengendefinition.....	251
Bild 59 — VC-Mengendefinition	252
Bild 60 — TC-Mengendefinition.....	253
Bild 61 — CFM-Mengendefinition.....	254
Bild 62 — PE-Mengendefinition.....	255
Bild 63 — Uhrenmengendefinition	255

Bild 64 — Zustandsmaschinen-Mengendefinition	256
Bild 65 — Allgemeines VC Nachrichtenformat	307
Bild 66 — Protokoll für das Lesen von Dateien	309
Bild 67 — Protokoll für den MLI Ferndownload	310
Bild 68 — Allgemeines SMLI Nachrichtenformat	336
Bild 69 — Allgemeines TC Nachrichtenformat	342
Bild 70 — Allgemeines MLI Nachrichtenformat	342
Bild 71 — Format der optionalen Parameterelemente	343
Bild 72 — Format von „Request PBIT Result“	345
Bild 73 — Format von „Reply PBIT Result“	345
Bild 74 — Format von „Request CFM Status“	346
Bild 75 — Format von „Reply CFM Status“	346
Bild 76 — Format von „Request CFM Info“	349
Bild 77 — Format von „Reply CFM Info“	349
Bild 78 — Format von „Test Message“	350
Bild 79 — Format von „Test Message Acknowledge“	350
Bild 80 — Format von „Request IBIT Start“	351
Bild 81 — Format von „Reply IBIT Start“	351
Bild 82 — Format von „Request IBIT Result“	352
Bild 83 — Format von „Reply IBIT Result“	352
Bild 84 — Format von „Load Image“	353
Bild 85 — Format von „Load Image Acknowledge“	355
Bild 86 — Format von „Load Routing Table“	356
Bild 87 — Format von „Load Routing Table Acknowledge“	361
Bild 88 — Format von „Load Time Configuration“	362
Bild 89 — Format von „Load Time Configuration Acknowledge“	365
Bild 90 — Format von „Request AGT“	366
Bild 91 — Format von „Reply AGT“	367
Bild 92 — Format von Ready_For_ALT_Synchro	367
Bild 93 — Format von Start_ALT_Synchro	368
Bild 94 — Format von „Request ALT“	369
Bild 95 — Format von „Reply ALT“	369

	Seite
Bild 96 — Format von „Request AGT ALT“	370
Bild 97 — Format von „Reply AGT ALT“	370
Bild 98 — Format von „Load Network Configuration“	371
Bild 99 — Format des NSM-Schaltbefehls	373
Bild 100 — Format für den NSM-Verbindungsbefehl	373
Bild 101 — Format für den NSM-Rücksetzbefehl	373
Bild 102 — Format für den NSM-Ausführungsbefehl.....	374
Bild 103 — Format von „Load Network Configuration Acknowledge“	374
Bild 104 — Format von „Request Network Status“	375
Bild 105 — Format von „Reply Network Status“	375
Bild 106 — Format von „Load Power Switches Configuration“	377
Bild 107 — Format des PCM-Schaltbefehls	378
Bild 108 — Format für den Leistungsschalterbefehl	378
Bild 109 — Format für das Zurücksetzen des Leistungsschalters.....	378
Bild 110 — Format von „Power Switch Configuration Acknowledge“	379
Bild 111 — Format von „Request Power Switches Status“	380
Bild 112 — Format von „Reply Power Switches Status“	380
Bild 113 — Allgemeines Protokoll für die CFM-Ressourcenverwaltung.....	383
Bild 114 — Allgemeines Protokoll für die Downloadverwaltung	385
Bild 115 — Allgemeines Protokoll für das Zeitmanagement.....	387
Bild 116 — Format von „Load Network Configuration“	392
Bild A.1 — Graphikkonzept.....	443

Tabellen

Tabelle 1 — Unabhängigkeit der Softwareschichten	13
Tabelle 2 — CBIT-Betriebsarten	24
Tabelle 3 — Routing-Informationen und Datentransfer.....	30
Tabelle 4 — Primitive IDL-Typen	39
Tabelle 5 — Zusammengesetzte IDL-Typen.....	40
Tabelle 6 — Leistungsschaltdienste.....	51
Tabelle 7 — Schichten, Prozessklassen und genormte Schnittstellen.....	60
Tabelle 8 — Liste der SMOS-Dienste für RE-CM	68

Tabelle 9 — Liste der SMOS-Dienste für RE-HM	68
Tabelle 10 — Liste der SMOS-Dienste für RE-FM	69
Tabelle 11 — Liste der SMOS-Dienste für RE-SM	70
Tabelle 12 — Liste der SMOS-Dienste für IA-CM	70
Tabelle 13 — Liste der SMOS-Dienste für IA-FM.....	71
Tabelle 14 — Liste der SMOS-Dienste für AC-FM	72
Tabelle 15 — Übergang von Thread-Zuständen.....	74
Tabelle 16 — Bedingung für den Zustandsübergang.....	75
Tabelle 17 — Eigenschaften der Zeitdienste	85
Tabelle 18 — Ressourcenparameter der Grundressourcen-Entitäten.....	103
Tabelle 19 — Kritikalitätsklassen der APOS-Dienste	104
Tabelle 20 — Definition der Sicherheitseinschränkungen.....	106
Tabelle 21 — APOS-Dienste	108
Tabelle 22 — APOS-Dateisucharten (Seek Mode).....	164
Tabelle 23 — MOS-Kerndienste	173
Tabelle 24 — Spezifische MOS-Borddienste	214
Tabelle 25 — Kundenspezifische MOS-Erweiterungsdienste.....	229
Tabelle 26 — Überblick über alle SMBP-Dienste	245
Tabelle 27 — Als ID_ITEM beschriebene Bezeichner	247
Tabelle 28 — Abbildung der EBNF-Spezifikation auf das RTBP-Konzept.....	248
Tabelle 29 — Überblick über alle SMOS-Dienste	267
Tabelle 30 — MLI-Download-Arten	299
Tabelle 31 — OLI-Dienste	311
Tabelle 32 — Liste der GLI-Dienste	314
Tabelle 33 — Liste der SMLI-Dienste.....	335
Tabelle 34 — MLI Dienste	344
Tabelle 35 — Definition der Nutzinformationfelder von „Reply PBIT Status“	345
Tabelle 36 — Definition der Nutzinformationfelder von „Reply CFM Status“	347
Tabelle 37 — Für GENERIC_CFM spezifische Erweiterung der Nutzinformationen	347
Tabelle 38 — Für NOT_GENERIC_CFM spezifische Erweiterung der Nutzinformationen.....	349
Tabelle 39 — Definition des Nutzinformationfeldes von „Reply CFM Info“	350
Tabelle 40 — Definition des Nutzinformationfeldes von „Reply IBIT Status“	351

Tabelle 41 — Definition der Nutzinformatiionsfelder von „Reply IBIT Result“	352
Tabelle 42 — Definition der Nutzinformatiionsfelder von „Load Image“	353
Tabelle 43 — Definition der Nutzinformatiionsfelder von „Load Image Acknowledge“	355
Tabelle 44 — Definition der Nutzinformatiionsfelder von „Load Routing Table“	356
Tabelle 45 — Datendefinition von „Load Routing Table“	357
Tabelle 46 — Datendefinition für die Schnittstellenkonfiguration.....	358
Tabelle 47 — Datendefinition für die Transferkonfiguration	359
Tabelle 48 — Datendefinition für die Protokollkonfiguration.....	360
Tabelle 49 — Datendefinition für „Destroy Transfer“	360
Tabelle 50 — Definition der Nutzinformatiionsfelder von „Load Routing Table Acknowledge“	361
Tabelle 51 — Definition der Nutzinformatiionsfelder von „Load Time Configuration“	362
Tabelle 52 — Datendefinition von „Load Time Configuration“	363
Tabelle 53 — Datendefinition für die Uhrenkonfiguration	363
Tabelle 54 — Datendefinition für die Konfiguration der föderierten Uhr	365
Tabelle 55 — Definition der Nutzinformatiionsfelder von „Load Time Configuration Acknowledge“	365
Tabelle 56 — Definition der Nutzinformatiionsfelder von „Reply AGT“	367
Tabelle 57 — Definition der Nutzinformatiionsfelder von Start_ALT_Synchro.....	368
Tabelle 58 — Definition der Nutzinformatiionsfelder von „Reply ALT“	369
Tabelle 59 — Definition der Nutzinformatiionsfelder von „Reply AGT ALT“	370
Tabelle 60 — Definition der Nutzinformatiionsfelder von „Load Network Configuration“	372
Tabelle 61 — Kodierung des Feldes für den NSM-Schaltbefehl	373
Tabelle 62 — Kodierung der Nutzinformatiionsfelder von „Load Network Configuration Acknowledge“	374
Tabelle 63 — Kodierung des Nutzinformatiionsfeldes von „Request Network Status“	375
Tabelle 64 — Kodierung der Nutzinformatiionsfelder von „Reply Network Status“	376
Tabelle 65 — Kodierung der Nutzinformatiionsfelder von „Load Power Switches Configuration“	377
Tabelle 66 — Kodierung des Feldes für den PCM-Schaltbefehl	378
Tabelle 67 — Kodierung der Nutzinformatiionsfelder von „Power Switch Configuration Acknowledge“	379
Tabelle 68 — Kodierung der Nutzinformatiionsfelder von „Reply Power Switches Status“	380
Tabelle 69 — Grundlegende ganzzahlige IDL-Typen	392
Tabelle 70 — Matrix für die Schnittstellenübereinstimmung	435
Tabelle 71 — Matrix für die Dienstübereinstimmung	435

	Seite
Tabelle A.1 — ASAAC-Graphiksprache	444
Tabelle A.2 — In Tabelle A.1 verwendete Kürzel.....	447
Tabelle A.3 — Hilfsfunktionen.....	448
Tabelle A.4 — Videothekfunktionen	448
Tabelle A.5 — Texturformate.....	450

Vorwort

Dieses Dokument (EN 4660-005:2011) wurde vom Verband der Europäischen Luft-, Raumfahrt- und Verteidigungsindustrie – Normung (ASD-STAN) erstellt.

Nachdem Umfragen und Abstimmungen entsprechend den Regeln dieses Verbandes durchgeführt wurden, hat diese Norm die Zustimmung der nationalen Verbände und offiziellen Behörden der Mitgliedsländer der ASD erhalten, bevor sie CEN vorgelegt wurde.

Diese Europäische Norm muss den Status einer nationalen Norm erhalten, entweder durch Veröffentlichung eines identischen Textes oder durch Anerkennung bis November 2011, und etwaige entgegenstehende nationale Normen müssen bis November 2011 zurückgezogen werden.

Es wird auf die Möglichkeit hingewiesen, dass einige Texte dieses Dokuments Patentrechte berühren können. CEN [und/oder CENELEC] sind nicht dafür verantwortlich, einige oder alle diesbezüglichen Patentrechte zu identifizieren.

Entsprechend der CEN/CENELEC-Geschäftsordnung sind die nationalen Normungsinstitute der folgenden Länder gehalten, diese Europäische Norm zu übernehmen: Belgien, Bulgarien, Dänemark, Deutschland, Estland, Finnland, Frankreich, Griechenland, Irland, Island, Italien, Kroatien, Lettland, Litauen, Luxemburg, Malta, Niederlande, Norwegen, Österreich, Polen, Portugal, Rumänien, Schweden, Schweiz, Slowakei, Slowenien, Spanien, Tschechische Republik, Ungarn, Vereinigtes Königreich und Zypern.

0 Einleitung

0.1 Zweck

Dieses Dokument wurde auf der Grundlage der zweiten Phase des ASAAC-Programms (ASAAC Phase II Contract 97/86.028) erarbeitet.

Zweck des ASAAC-Programms sind die Definition und die Validierung einer Reihe von Standards, Konzepten und Leitlinien für offene A3-Architekturen (Advanced Avionics Architectures), mit denen die drei Hauptzielsetzungen des Programms erfüllt werden sollen. Die im Rahmen des Programms erarbeiteten Standards, Konzepte und Leitlinien sollen sowohl für neue Flugzeuge als auch für Programmaktualisierungen aus dem Jahr 2005 anwendbar sein.

Die drei Hauptzielsetzungen des ASAAC-Programms sind:

1. die Senkung der Kosten während des Lebenszyklus;
2. die Verbesserung der Leistung bei der Erfüllung der Mission;
3. die Verbesserung der Leistung während des Betriebs.

Die Standards sind als eine Gruppe von Dokumenten organisiert, die Folgendes umfassen:

- eine Reihe von vereinbarten Standards, die mit Hilfe eines Top-down-Ansatzes den Überblick über die Architektur aller Schnittstellen beschreiben, die zur Implementierung des Kerns von Avioniksystemen erforderlich sind;
- die Leitlinien für die Systemimplementierung durch Anwendung der Standards.

Es folgt ein Überblick über die Dokumentenhierarchie: *(das aktuelle Dokument ist im Bild hervorgehoben)*

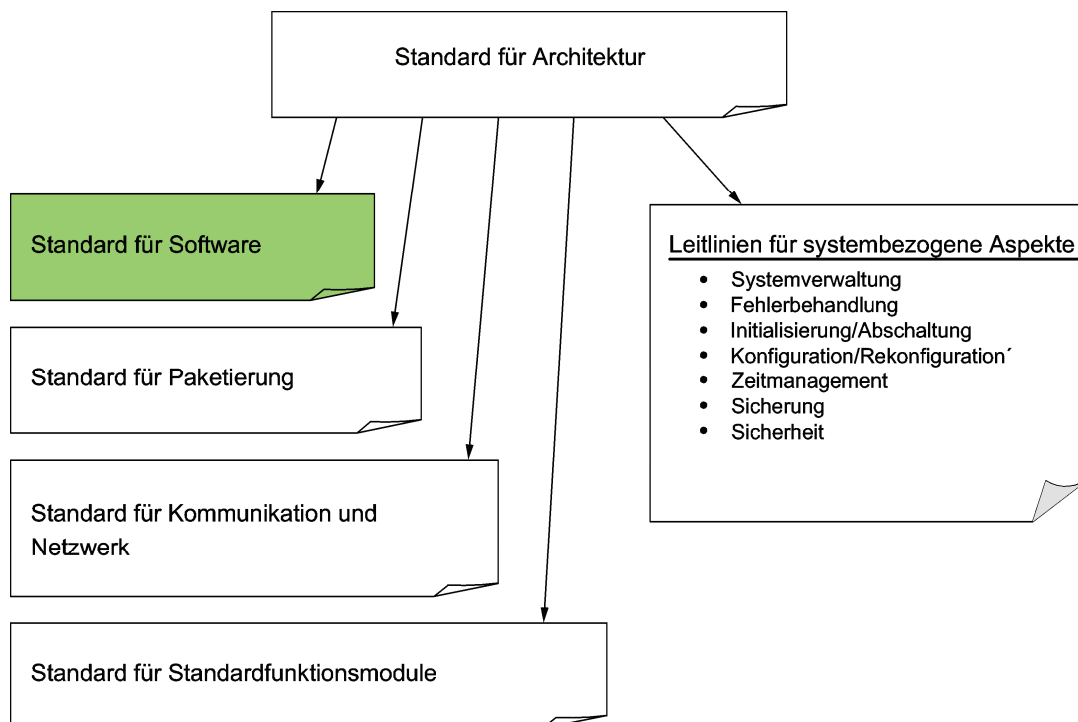


Bild 1 — Hierarchie der Dokumentation der ASAAC-Standards

0.2 Aufbau des Dokuments

Das vorliegende Dokument enthält die folgenden Abschnitte:

- Abschnitt 1: Anwendungsbereich;
- Abschnitt 2: Normative Referenzen;
- Abschnitt 3: Begriffe und Abkürzungen;
- Abschnitt 4: Systemfunktionen;
- Abschnitt 5: Definition der Software-Architektur;
- Abschnitt 6: Definition direkter Schnittstellen;
- Abschnitt 7: Definition logischer Schnittstellen;
- Abschnitt 8: Datentypdefinitionen;
- Abschnitt 9: Implementierungsspezifischer Zuschnitt;
- Anhang A: AGL.

1 Anwendungsbereich

Zweck der vorliegenden Norm ist die Festlegung einheitlicher Anforderungen an den Entwurf und die Entwicklung einer Software-Architektur für modulare Avioniksysteme nach dem ASAAC-Standard.

1.1 Überblick über die Software-Architektur

Die ASAAC-Software-Architektur basiert auf einem Dreischichtenstapel, wie in Bild 2 vereinfacht dargestellt.

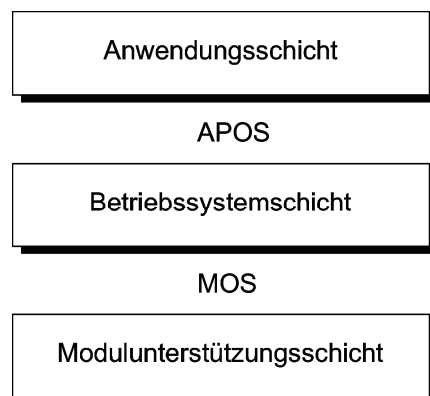


Bild 2 — Dreischichtige ASAAC-Software-Architektur

Jede Schicht kann im Hinblick auf ihre Abhängigkeit/Unabhängigkeit vom Flugzeugsystem und von der zugrunde liegenden Hardware beschrieben werden.

1.2.4 Generische Systemverwaltung (GSM)

Das GSM ist für die Verwaltung des Core-Processing verantwortlich (siehe 4.1.1 und 5.2.1). Diese Funktionalität ist in vier Bereiche unterteilt:

- Zustandsüberwachung;
- Fehlerbehandlung;
- Konfigurationsverwaltung;
- Sicherheitsmanagement.

1.2.5 Runtime-Blueprints (RTBP)

Die RTBP enthalten die Informationen (z. B. Prozessbeschreibung, Routinginformationen, Fehlerbehandlungsdaten), die zur Konfiguration und Verwaltung des Core-Processing, auf dem sie gehostet werden, erforderlich sind (siehe 5.3).

1.2.6 Modulunterstützungsschicht (MSL)

Die MSL kapselt die Einzelheiten der zugrunde liegenden Hardware und ermöglicht einen generischen, technologieunabhängigen Zugriff auf Low-Level-Ressourcen (siehe 5.1).

1.2.7 Schnittstelle Anwendung/Betriebssystem (APOS)

Die APOS ist eine direkte Schnittstelle, die die flugzeugabhängige Software (AL) von der flugzeugunabhängigen Software (OSL) trennt. Ihr Zweck ist es, die Prozesse in der AL mit einer genormten OS-unabhängigen Schnittstelle zu den vom OS bereitgestellten Diensten bereitzustellen und so die Portabilität und Wiederverwendung von Anwendungssoftware zu fördern (siehe 6.1).

1.2.8 Schnittstelle Modulunterstützung/Betriebssystem (MOS)

Die MOS ist eine direkte Schnittstelle, die die OSL von der hardwareabhängigen Software (MSL) trennt. Ihr Zweck ist es, die OS mit einer hardwareunabhängigen/technologisch transparenten Schnittstelle zu der in der MSL enthaltenen Funktionalität zu versehen. Daher lässt die MOS zu, dass dieselbe OSL-Software auf unterschiedlichen Implementierungen eines bestimmten CFM vorhanden ist, und zwar unabhängig von der zugrunde liegenden Hardware (siehe 6.2).

1.2.9 Schnittstelle Systemverwaltung/Blueprints (SMBP)

Diese direkte Schnittstelle, die in der OSL zwischen GSM und den Blueprints gekapselt ist, ermöglicht, dass Struktur und Implementierung der Blueprints ungenormt bleiben, während sie eine genormte Schnittstelle zu ihnen definiert (siehe 6.3).

1.2.10 Schnittstelle Systemverwaltung/Betriebssystem (SMOS)

Diese in der OSL gekapselte direkte Schnittstelle beschreibt die vom OS für das GSM bereitgestellten Dienste (siehe 6.4).

1.2.11 Logische OS-Schnittstelle (OLI)

Die OLI beschreibt die Kommunikation zwischen zwei OS-Instanzierungen im Hinblick auf die Kommunikation des virtuellen Kanals (VC) und die Datendarstellung (siehe 7.1).

1.2.12 Logische GSM-Schnittstelle (GLI)

Das GLI beschreibt die Kommunikation zwischen zwei GSM-Instanzierungen (siehe Abschnitt 0). Diese Kommunikation zwischen GSM erfolgt hierarchisch.

1.2.13 Logische Systemverwaltungsschnittstelle (SMLI)

Das SMLI normt ein VC-basiertes Kommunikationsprotokoll zwischen der AM und dem GSM. AM und GSM müssen miteinander zusammenarbeiten; dazu kommunizieren sie und synchronisieren sich über das SMLI (siehe 7.3).

1.2.14 Logische Modulschnittstelle (MLI)

Diese logische Schnittstelle (Kommunikationsprotokoll) definiert die logischen Interaktionen zwischen Modulen, sodass die Anforderungen an die Modulinteroperabilität und die Systembaubarkeit erfüllt werden (siehe 7.4).

2 Normative Verweisungen

Die folgenden zitierten Dokumente sind für die Anwendung dieses Dokuments erforderlich. Bei datierten Verweisungen gilt nur die in Bezug genommene Ausgabe. Bei undatierten Verweisungen gilt die letzte Ausgabe des in Bezug genommenen Dokuments (einschließlich aller Änderungen).

EN 4660-001, *Luft- und Raumfahrt — Modulare und offene Avionikarchitekturen — Teil 001: Architektur*

EN 4660-002, *Luft- und Raumfahrt — Modulare und offene Avionikarchitekturen — Teil 002: CFM*

EN 4660-003, *Luft- und Raumfahrt — Modulare und offene Avionikarchitekturen — Teil 003: Kommunikation/Netzwerk*

EN 4660-004, *Luft- und Raumfahrt — Modulare und offene Avionikarchitekturen — Teil 004: Paketierung*

ASAAC2-GUI-32450-001-CPG Issue 01, *Final Draft of Guidelines for System Issues*¹⁾

— Volume 1 — System Management

— Volume 2 — Fault Management

— Volume 3 — Initialisation and Shutdown

— Volume 4 — Configuration/Reconfiguration

— Volume 5 — Time Management

— Volume 6 — Security

— Volume 7 — Safety

Common Object Request Broker Architecture: Core Specification Version 3.0 — *Editorial update formal/02-12-06, OMG*

ISO/IEC 14977:1996, *Information technology — Syntactic metalanguage — Extended BNF*

ISBN 0-201-63276-4, *Open GL Reference Manual*

3 Begriffe und Abkürzungen

Für die Anwendung dieses Dokuments gelten die folgenden Begriffe und Abkürzungen.

1) Zum Zeitpunkt der Veröffentlichung dieser Norm in Vorbereitung.

3.1 Begriffe

In der vorliegenden Norm werden die Begriffe „muss/müssen“, „sollte/sollten“ und „darf/dürfen“ wie folgt verwendet:

- MUSS/MÜSSEN drückt eine verbindliche Anforderung dieser Norm aus;
- SOLLTE/SOLLTEN drückt eine Empfehlung oder einen Ratschlag zur Implementierung einer verbindlichen Anforderung der Norm aus. Es wird erwartet, dass diese Empfehlung bzw. dieser Ratschlag befolgt wird, sofern nicht gute Gründe dafür angegeben sind, dies nicht zu tun;
- DARF/DÜRFEN drückt eine zulässige Praktik oder Handlung aus. Dabei handelt es sich nicht um eine verbindliche Anforderung der Norm.

3.2 Abkürzungen

Abkürzung	Englisch	Deutsch
AC	Aircraft	Flugzeug/Luftfahrzeug
AGL	ASAAC Graphics Tag Language	ASAAC-Graphik-Tag-Sprache
AGT	Absolute Global Time	absolute globale Zeit
AL	Application Layer	Anwendungsschicht
ALT	Absolute Local Time	absolute lokale Zeit
AM	Application manager	Anwendungsmanager
APOS	Application to Operating System [Interface]	Schnittstelle Anwendung/Betriebssystem
ASAAC	Allied Standard Avionics Architecture Council	—
ATM	Asynchronous Transfer Mode	asynchrones Übermittlungsverfahren
BIT	Built-in Test	Selbsttest
BMC	Between Module Communication	Kommunikation zwischen Modulen
CBIT	Continous BIT	kontinuierlicher BIT
CDR	Common Data Representation	gemeinsame Datendarstellung
CFM	Common Functional Module	Standardfunktionsmodul
CM	Configuration Management	Konfigurationsverwaltung
COTS	Commercial Off The Shelf	seriengefertigte kommerzielle Produkte
CPU	Central Processing Unit	Zentraleinheit
DMC	Distributed Multicast Communication	verteilte Multicast-Kommunikation
DPM	Data Processing Module	Datenverarbeitungsmodul
EBNF	Extended Backus-Naur Form	erweiterte Backus-Naur-Form
EW	Electronic Warfare	elektronische Kampfführung
FC	Fibre Channel	Faserkanal
FIFO	First In First Out	—
FM	Fault Management	Fehlerbehandlung
GLI	Generic System Management Logical Interface	logische Schnittstelle der generischen Systemverwaltung
GSM	Generic System Management	generische Systemverwaltung
HM	Health Monitoring	Zustandsüberwachung
HW	Hardware	—
IA	Integration Area	Integrationsbereich

Abkürzung	Englisch	Deutsch
IBIT	Initiated BIT	eingeleiteter BIT
ID	Identification	Identifikation
IDL	Interface Definition Language	Schnittstellendefinitionssprache
IF	Interface	Schnittstelle
IMA	Integrated Modular Avionics	Integrierte Modulare Avionik
IMC	Intra Module Communication	Intramodulkommunikation
IPC	Intra Processor Communication	Intraprozessorkommunikation
IPEC	Intra PE Communication	Intra-PE-Kommunikation
LC	Logical Configuration	logische Kommunikation
LSB	Least Significant Byte	niederwertigstes Byte
MC	Master Clock	Hauptuhr
MLI	Module Logical Interface	logische Modulschnittstelle
MMM	Mass Memory Module	Massenspeichermodul
MOS	MSL to OS [Interface]	MSL/OS-Schnittstelle
MRC	Master Reference Clock	Haupt-Referenzuhr
MSB	Most Significant Byte	höchstwertiges Byte
MSL	Module Support Layer	Modulunterstützungsschicht
MSU	Module Support Unit	Modulunterstützungseinheit
N/A	Not Applicable	nicht anzuwenden
NC	Network Channel	Netzwerkkanal
NII	Network Independent Interface	netzwerkunabhängige Einheit
NIU	Network Interface Unit	Netzwerkschnittstelleneinheit
NSM	Network Support Module	Netzwerkunterstützungseinheit
NW	Network	Netzwerk
OLI	Operating System Logical Interface	logische Schnittstelle des Betriebssystems
OMG	Object Management Group	Objektverwaltungsgruppe
OS	Operating System	Betriebssystem
OSL	Operating System Layer	Betriebssystemschiicht
PBIT	Power-up BIT	BIT bei Einschalten
PCM	Power Conversion Module	Stromrichtermodul
PE	Processing Element	Verarbeitungselement
PSE	Power Supply Element	Stromversorgungselement
PU	Processing Unit	Verarbeitungseinheit
QoS	Quality of Service	Dienstgüte
RE	Resource Element	Ressourcenelement
RC	Remote Clock	externe Uhr
RF	Radio Frequency	Hochfrequenz
RLT	Relative Local Time	relative lokale Zeit
RTBP	Runtime Blueprints	Laufzeit-Blueprints
RU	Routing Unit	Routing-Einheit
SCU	Switch Control Unit	Schaltsteuereinheit

Abkürzung	Englisch	Deutsch
SM	Security Management	Sicherheitsmanagement
SMBP	System Management to Blueprints Interface	Schnittstelle Systemverwaltung/Blueprints
SMLI	System Management Logical Interface	logische Schnittstelle der Systemverwaltung
SMOS	System Management to OS Interface	Schnittstelle Systemverwaltung/Betriebssystem
SPM	Signal Processing Module	Signalverarbeitungsmodul
SW	Software	—
TC	Transfer Connection	Transferverbindung
TLS	Three Layer Stack	Dreischichtenstapel
VC	Virtual Channel	virtueller Kanal
VL	Video Library	Videothek

4 Systemfunktionen

Der vorliegende Abschnitt beschreibt den Systemkontext und die Abbildung von Systemfunktionen auf Komponenten der Software-Architektur.

4.1 Systemverwaltungsfunktion

ANMERKUNG Zu den Anforderungen an die Systemverwaltung, einschließlich ausführlicher Beispiele, siehe die entsprechenden Teile von ASAAC2-GUI-32450-001-CPG Issue 01.

Die Systemverwaltung ist verantwortlich für die Verwaltung des ASAAC-Systems während der Initialisierung, während aller Betriebsphasen im Flug und am Boden und während des Herunterfahrens des Systems bis zur Stromabschaltung. Die Aufgaben der Systemverwaltung umfassen somit:

- Steuerung der Systeminitialisierungs-, Rekonfigurations- und Abschaltprozesse;
- Identifizierung, Maskierung, Filterung und Lokalisierung von Fehlern;
- Bereitstellung von sicherheitsbezogenen Diensten.

Die Systemverwaltung setzt sich aus zwei Funktionen auf der Anwendungsschicht und der OSL des dreischichtigen ASAAC-Modells zusammen (Bild 3):

- der Anwendungsmanagementfunktion (AM): flugzeugabhängig, HW-unabhängig,
- der GSM-Funktion (GSM): flugzeugabhängig, HW-unabhängig.

Die zugrunde liegenden Prinzipien dieser Architektur sind die Trennung zwischen der Hardware und flugzeugabhängigen Schichten sowie zwischen Avionikfunktionen, die durch Funktionsanwendungen und durch Systemverwaltungsfunktionen dargestellt werden.

Die Systemverwaltungsfunktion ist hierarchisch auf drei Arten von Ebenen organisiert (Bild 4), die folgende Funktionalitäten abdecken:

- Luftfahrzeugebene (AC-Ebene),
- Integrationsbereichsebene (IA-Ebene),
- Ressourcenelement-Ebene (RE-Ebene).

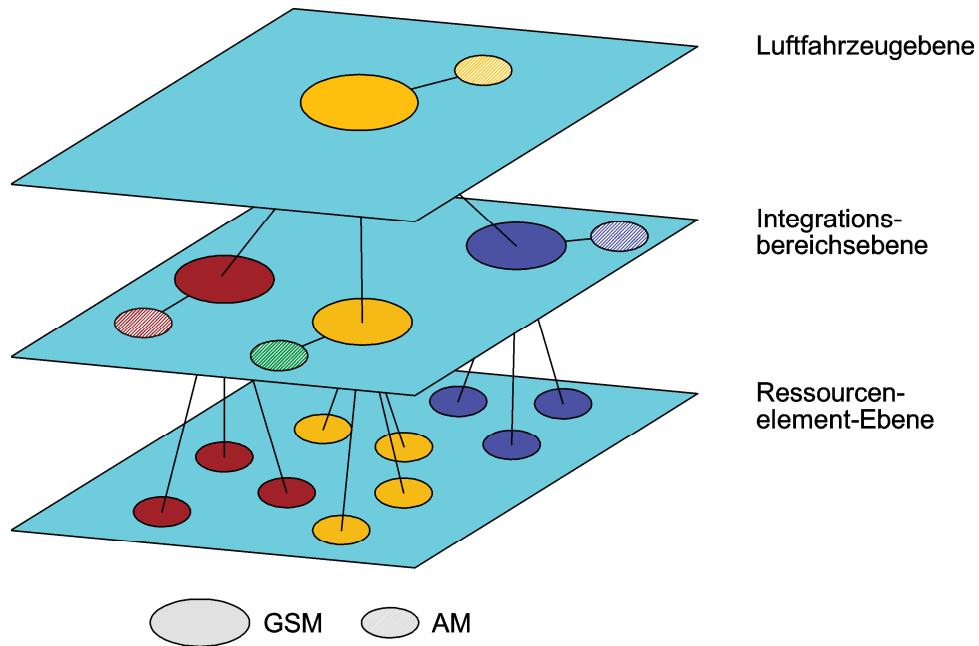


Bild 4 — Hierarchische Organisation der Systemverwaltung

Jede Ebene weist spezielle Merkmale auf:

AC-Ebene:

- die AC-Ebene ist eine einzelne Systemverwaltungsentität, die für die Steuerung/Überwachung des gesamten IMA-Kerns verantwortlich ist.

IA-Ebene:

- die IA-Ebene ist eine logische Gruppierung eng verzahnter Funktionsanwendungen und ihrer Ressourcen. Diese Gruppierung braucht nicht statisch zu sein, sondern kann dynamisch während einer Mission erzeugt und gelöscht werden.
- jeder IA steuert ein oder mehrere RE.
- IA dürfen intern hierarchisch organisiert sein, d. h. ein System kann eine oder mehrere IA-Ebenen aufweisen. In diesem Fall muss der IA der niedrigsten Ebene ein oder mehrere PE steuern.
- ein System kann auch so ausgelegt sein, dass es keine IA-Ebene aufweist. In diesem Fall gibt es eine AC-Ebene und eine oder mehrere RE-Ebenen.

RE-Ebene:

- das Ressourcenelement ist die niedrigste adressierbare Ebene in der Systemverwaltungshierarchie; sie ist für die Verwaltung der Funktionalität eines einzelnen Verarbeitungselements (PE) verantwortlich.

4.1.1 GSM-Funktion

Die GSM-Funktion GSM ist verantwortlich für die Verwaltung und Steuerung aller Ressourcen und die Verwaltung des Verhaltens des ASAAC-Systems mit Hilfe von RTBP (siehe 5.3).

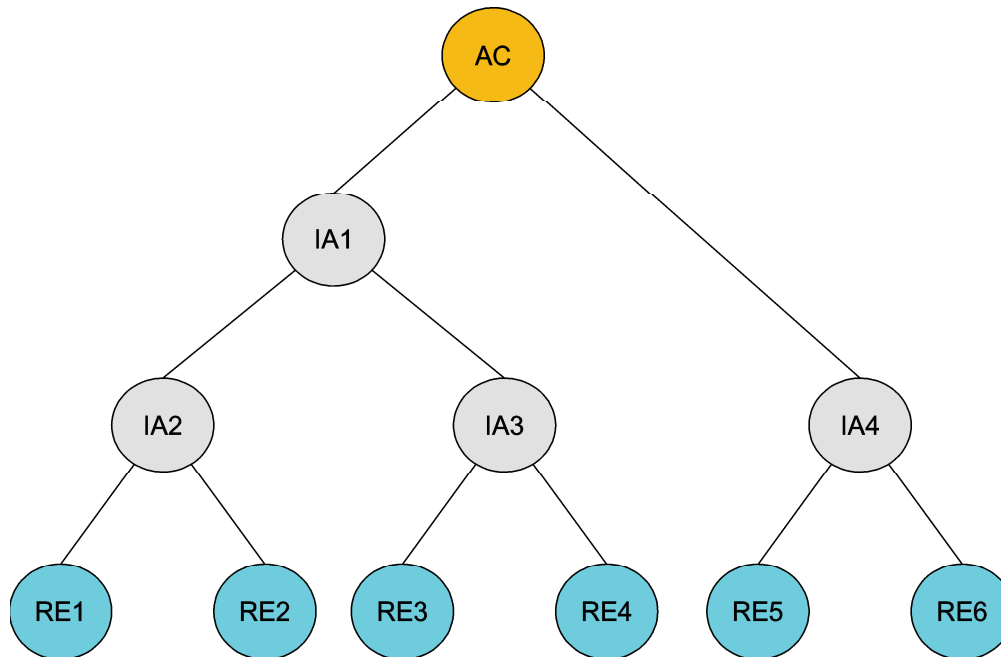


Bild 5 — GSM-Aufgliederung für RE-Verwaltung (Beispiel)

Funktionen:

Das GSM umfasst vier Funktionen:

- Konfigurationsverwaltung:
Einrichtung der Ausgangskonfiguration, Rekonfigurationsverwaltung.
- Zustandsüberwachung:
Überwachung des Zustands, Erfassung, Filterung und Übertragung von Fehlern an die FM.
- Fehlerbehandlung:
Maskierung, Filterung und Lokalisierung von Fehlern, einschließlich Verarbeitung von behebenden Maßnahmen.
- Sicherheitsmanagement:
Implementierung der Sicherheitsvorgaben für das System: Authentisierung, Entschlüsselung und Verschlüsselung.

Hierarchische Organisation:

Die folgenden Bilder veranschaulichen mögliche Abbildungen von Ressourcen auf die Systemverwaltungshierarchie:

- RE-Verwaltung (Bild 5): Ein AC-Manager steuert 2 IA-Manager, IA1 und IA4; IA1 steuert die IA-Manager IA2 und IA3; IA2, IA3 und IA4 steuern jeweils 2 RE.

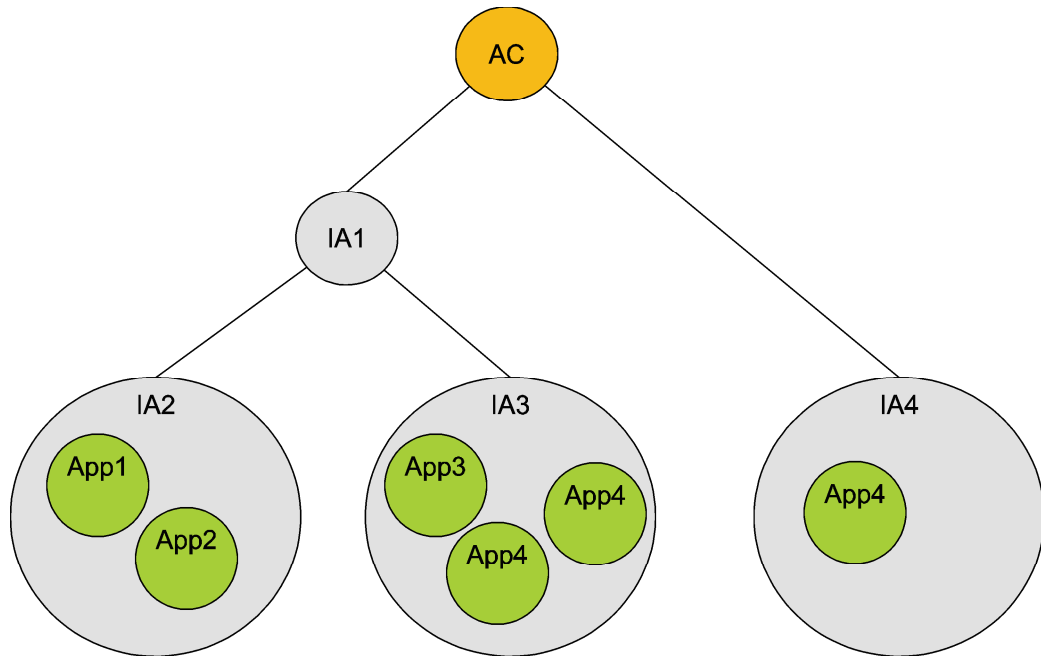


Bild 6 — IA-Anwendungssteuerung (Beispiel)

- Anwendungskonfigurationssteuerung (Bild 6): Ein AC-Manager steuert IA1 und IA4; IA1 steuert IA2 und IA3; IA2 steuert die Anwendungen App1 und App2; IA3 steuert die Anwendungen App3 und App4 (redundant); IA4 steuert die redundante Anwendung App4;

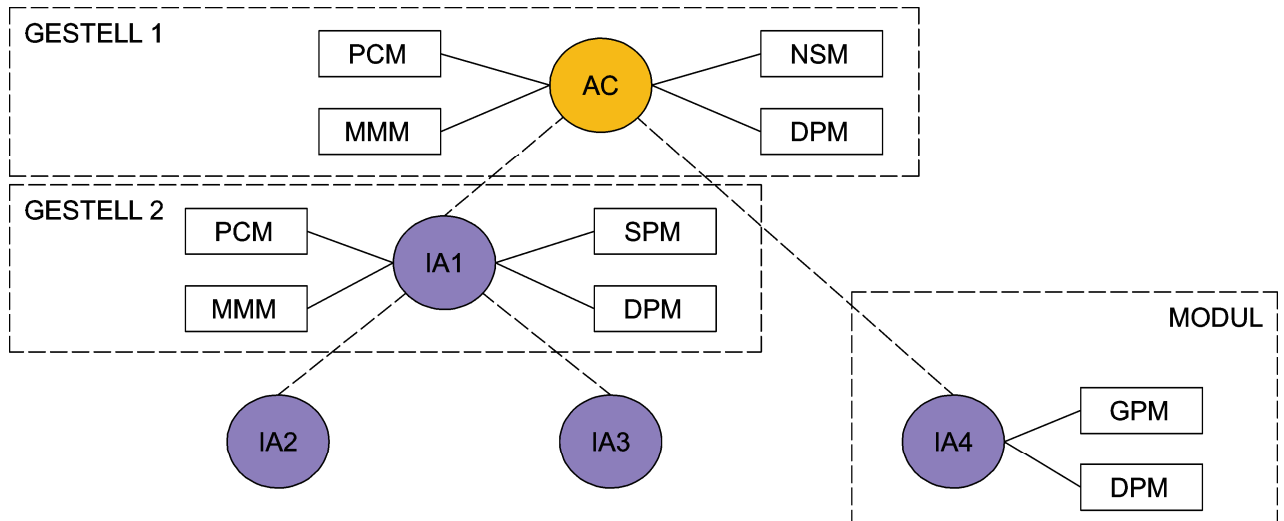


Bild 7 — GSM-Aufgliederung für Modulverwaltung (Beispiel)

- Anwendungskonfigurationssteuerung (Bild 7): Ein AC-Manager steuert IA1 und IA4; IA1 steuert IA2 und IA3; IA2 steuert die Anwendungen App1 und App2; IA3 steuert die Anwendungen App3 und App4 (redundant); IA4 steuert die redundante Anwendung App4.

Konfigurationsdaten:

Die Konfigurationsdaten werden von den RTBP über die SMBP bezogen. Die Rekonfiguration wird durch über die SMBP bezogene dedizierte Sequenzen definiert.

Initialisierung und Abschaltung:

Initialisierung und Abschaltung erfolgen auf drei unterschiedlichen Ebenen:

- Anwendung,
- System,
- Modul.

4.1.2 AM-Funktion

Die AM-Funktion ist verantwortlich für die Verwaltung und Steuerung aller AC-abhängigen Funktionen (Funktionsanwendungen) auf der Anwendungsschicht (AL). Sie dient als Schnittstelle zwischen den Funktionsanwendungen und einer dedizierten Instanzierung des GSM.

Hierarchische Organisation:

Das AM sollte sich nur auf der AC- und IA-Ebene befinden, da die RE-Ebene ressourcenorientiert ist, die AC- und IA-Ebene hingegen funktionsorientiert sind.

Ein Beispiel für die hierarchische Organisation der AM, die die Zuweisung von Funktionsanwendungen zu IA zeigt, ist in Bild 8 dargestellt:

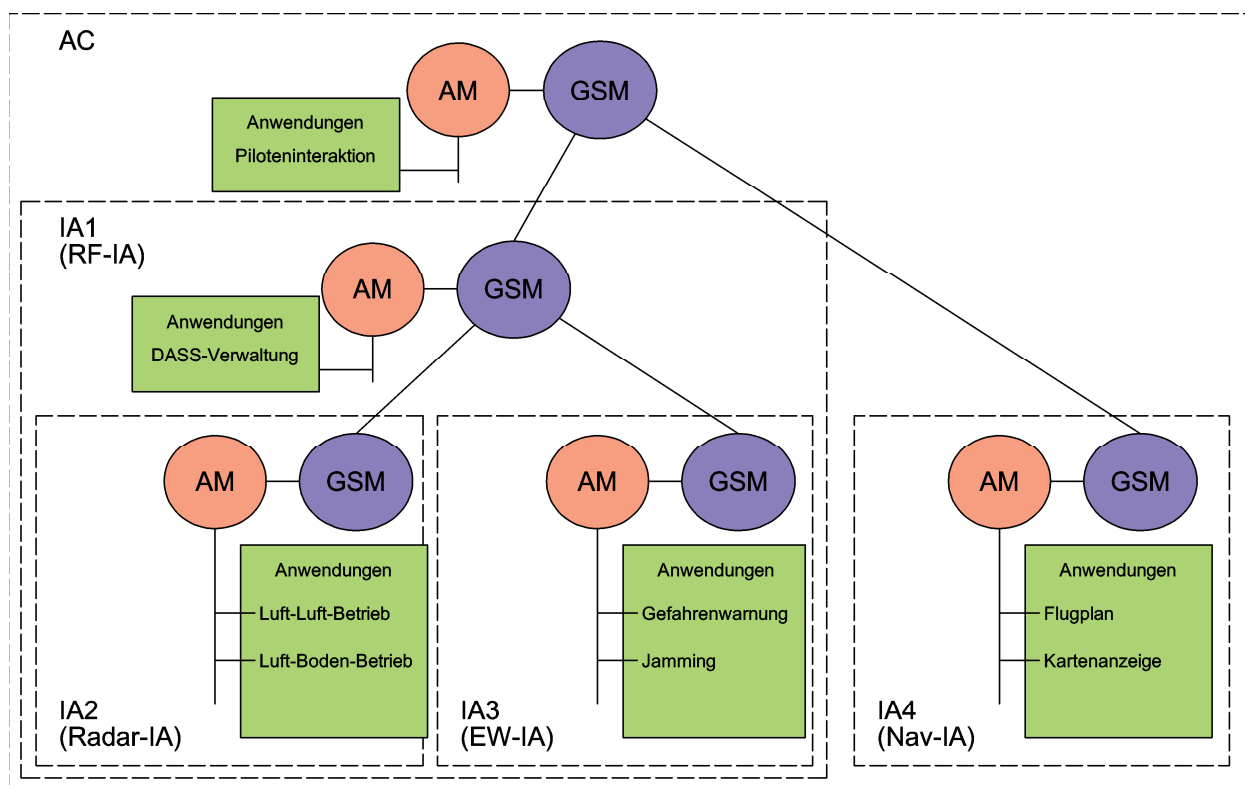


Bild 8 — Hierarchische Organisation des AM (Beispiel)

Interne Schnittstellen:

Die genormte interne Schnittstelle des AM ist die logische Systemverwaltungsschnittstelle (SMLI). Das SMLI schließt ein Anfrage-Antwort-Protokoll für die Änderung der logischen Konfiguration ein.

Externe Schnittstellen:

Es gibt keine genormten externen AM-Schnittstellen. Sämtliche externen Schnittstellen sind anwendungsabhängig.

4.1.3 Fehlerverarbeitung

Bei Systemen, die dem ASSAC-Standard entsprechen, ist es erforderlich, dass die Software-Entwickler den Funktionsanwendungscode so schreiben, dass dieser eine Schnittstelle mit dem zugrunde liegenden Betriebssystem bildet, wobei die genormten Dienstaufrufe verwendet werden, die die APOS-Schnittstelle umfassen (6.1). Während der Laufzeit ist es jedoch möglich, dass ein APOS-Dienst nicht korrekt funktioniert und einen Fehlerzustand an den aufrufenden Anwendungsprozess zurückgibt. Dies kann auf einen tatsächlichen Fehler im zugrunde liegenden System zurückzuführen sein oder auf eine Fehlverwendung der APOS-Schnittstellen selbst (z. B. Senden eines Semaphors, bevor dieser erzeugt wurde). In jedem Fall wird der Fehler durch einen Standardprozess (siehe ASAAC2-GUI-32450-001-CPG Issue 01) verarbeitet, bei dem die genaue Fehleridentifizierung an die Zustandsüberwachungsfunktion innerhalb des GSM weitergegeben wird. Jede Fehlerverarbeitung muss den Entscheidungen der Fehlerbehandlungsfunktion unterliegen. Bei der Verarbeitung des Fehlers darf die Fehlerbehandlungsfunktion die Fehlerverarbeitung zurück an einen funktionalen Anwendungsprozess delegieren, indem sie den Fehlerbehandler-Thread des Anwendungsprozesses aufruft. In diesem Fall muss dieser Fehlerbehandler-Thread auf die vollständigen Fehlerinformationen zugreifen können.

Die Fehlerinformationen müssen für die Anwendung selbst zugänglich sein, dürfen aber nur für Debugging genutzt werden.

Ausnahmen sind Timeouts und Ressourcen, die von der Anwendung verwaltet werden. Zu beachten ist jedoch, dass funktionale Anwendungsprozesse Situationen handhaben müssen, in denen ein aufgerufener APOS-Dienst die Zeit überschritten hat. In diesem Fall muss die Anwendung, die einen Dienst aufruft, durch eine Rückmeldung darüber informiert werden.

4.1.4 Selbsttest

Der BIT-Dienst stellt die Fähigkeit bereit, Modul-Selbsttests auszuführen und die entsprechenden Ergebnisse zu lesen. Die Selbsttestkomponente bietet Zugriff auf sämtliche BIT-Routinen, die auf dem Modul verfügbar sind. Es gibt drei unterschiedliche Selbsttestarten:

- Selbsttest beim Einschalten (PBIT),
- kontinuierlicher Selbsttest (CBIT),
- eingeleiteter Selbsttest (IBIT).

Das Betriebssystem stellt dem GSM Dienste im Zusammenhang mit der BIT-Verwaltung an der SMOS-Schnittstelle zur Verfügung, die mit Diensten an der MOS-Schnittstelle gepaart sind:

- Get PBIT Result: ruft das gespeicherte PBIT-Ergebnis ab,
- Start CBIT: führt die CBIT-Verarbeitung aus und springt anschließend zurück. Erlaubt das Ausführen einer bestimmten Testart oder aller Tests,
- Get CBIT Result: ruft das CBIT-Ergebnis ab,
- Start IBIT: startet die IBIT-Verarbeitung,
- Get IBIT Result: ruft das gespeicherte IBIT-Ergebnis ab.

4.1.4.1 Selbsttest beim Einschalten (PBIT)

PBIT wird verwendet, um den Zustand der Modulhardware als Teil des Bootprozesses zu überprüfen. Die Tests werden eigenständig als Teil der MSL ausgeführt, bevor von außerhalb des Moduls eine Steuerung eingreift. Das Ergebnis dieser Tests wird in der MSL aufgezeichnet, um dann von einem GSM auf einem Steuermodul über die MLI abgerufen werden zu können. Es steht auch über einen MOS/SMOS-Aufruf an das lokale GSM zur Verfügung.

4.1.4.2 Kontinuierlicher Selbsttest (CBIT)

CBIT werden verwendet, um bei Normalbetrieb fortlaufend den Zustand des Moduls zu überprüfen. CBIT funktionieren unterbrechungsfrei. Die Tests können wie folgt ausgeführt werden:

- entweder autonom, wenn zur Ausführung keine Prozessorunterstützung erforderlich ist,
- oder unter dem Befehl des GSM, wenn zur Ausführung eine Prozessorunterstützung erforderlich ist.

Die Ergebnisse können durch zwei Mechanismen bezogen werden:

- Rückruf,
- Umfrage, entweder als Teil des Aufrufmechanismus oder als separater Aufruf.

Es folgt eine Beschreibung der verschiedenen Kombinationen des CBIT-Verhaltens.

Tabelle 2 — CBIT-Betriebsarten

Ausführungs- verfahren	Ergebnis- verfahren	Verhalten
Autonom	Rückruf	CBIT wird eigenständig ausgeführt und erfordert keine Steuerung von außerhalb der MSL. Ein Nichtbestehen wird der OSL mittels Rückruf angezeigt. Daraufhin wird der Dienst „getCbitResult“ verwendet, um die ausführlichen Informationen zum Nichtbestehen abzurufen.
Autonom	Umfrage	CBIT wird eigenständig ausgeführt und erfordert keine Steuerung von außerhalb der MSL. Bei einem Nichtbestehen wird das Ergebnis intern in der MSL gespeichert. Die OSL erhält keine Mitteilung. Anschließend wird „getCbitResult“ periodisch verwendet, um Informationen zum Nichtbestehen abzurufen. Ist es zu keinem Versagen gekommen, werden keine Maßnahmen getroffen. Bei einem Nichtbestehen werden die ausführlichen Informationen dazu zurückgegeben.
Auf Befehl	Rückruf	Die CBIT-Ausführung wird von der OSL gesteuert. Ein Nichtbestehen wird der OSL mittels Rückruf angezeigt. Daraufhin wird „getCbitResult“ verwendet, um die ausführlichen Informationen zum Nichtbestehen abzurufen.
Auf Befehl	Umfrage	Die CBIT-Ausführung wird von der OSL gesteuert. Die Zeit, die zum Ausführen des CBIT bei jedem Aufruf des Dienstes „startCbit“ zur Verfügung steht, ist MSL-spezifisch. Wird ein Nichtbestehen erkannt, wird „getCbitResult“ verwendet, um die ausführlichen Informationen dazu abzurufen.

4.1.4.3 Eingeleiteter Selbsttest (IBIT)

IBIT wird verwendet, um den Zustand der Modulhardware als Teil des Fehlerbehandlungsprozesses zu überprüfen. Es wird eine umfassende Prüfung des Moduls durchgeführt, um bei der Fehlerlokalisierung zu helfen. Die Tests können aus der Ferne ausgeführt werden, gesteuert von einem GSM auf einem Steuermodul über die MLI, oder über einen MOS/SMOS-Aufruf (startIbit) von dem lokalen GSM, sofern verfügbar. Der IBIT-Betrieb kann zerstörend wirken. Das heißt, dass die aktuelle Konfiguration des Moduls nach Abschluss der Tests nicht garantiert werden kann. Daher ist darauf zu achten, dass sichergestellt wird, dass das System bei Ausführung eines IBIT nicht beeinträchtigt wird. Außerdem sollte die Anwendung eines zerstörenden Tests auf den Aufruf über die MLI beschränkt sein. Das Ergebnis dieser Tests wird in der MSL aufgezeichnet, um durch einen GSM auf einem Steuermodul über die MLI oder über einen MOS/SMOS-Aufruf (getIbitResult) von dem lokalen GSM, sofern diese den IBIT gestartet hat, abgerufen werden zu können.

4.2 Kommunikation

4.2.1 ASAAC-Kommunikationsmodell

4.2.1.1 Kurzbeschreibung

Der ASAAC-Kommunikationsstapel (siehe Bild 9) muss unterstützt werden durch:

- VC (bereitgestellt von der OSL);
- Transferverbindungen (TC) (bereitgestellt von der MSL, hardwareunabhängig);
- Netzwerkkanälen (NC) (bereitgestellt von der MSL, hardwareabhängig).

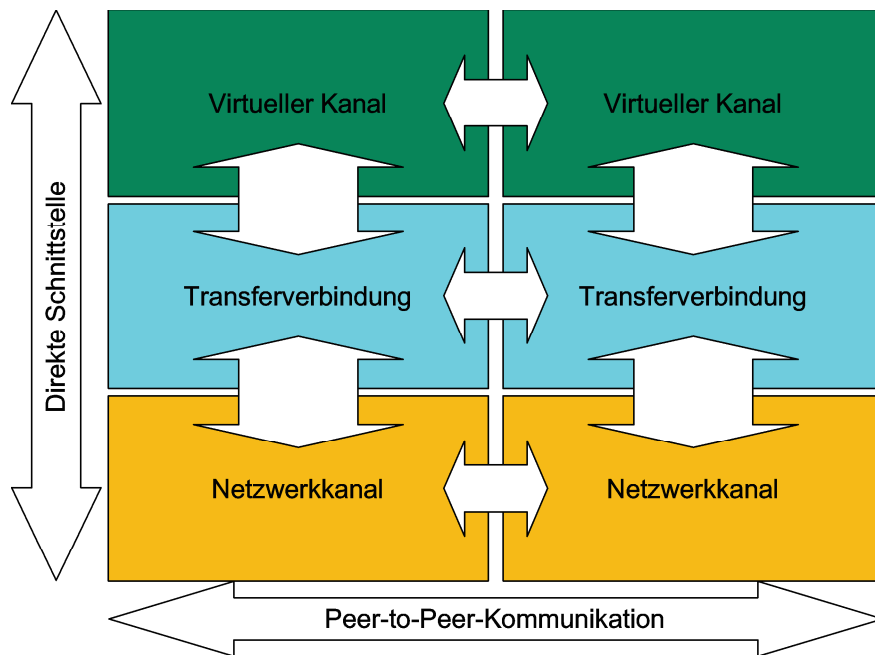


Bild 9 — Der ASAAC-Kommunikationsstapel

Die ASSAC-Kommunikation muss folgende Kommunikationen unterstützen:

- ein Sender an einen Empfänger (1:1);
- Multicast (ein Sender an n Empfänger (1: N); der Fall „ein Sender an einen Empfänger“ ist eine Teilmenge des oben genannten Falls (1:1));
- verteiltes Multicast (gilt für Signalverarbeitungsanwendungen ($M:N$)).

4.2.1.2 VC

Die Interprozesskommunikation basiert auf virtuellen Kanälen. Diese weisen folgende Eigenschaften auf:

- Unidirektional,
- nachrichtenorientiert (d. h. eine Nachrichtendefinition ist einem VC zuwiesen),
- verwaltet durch OSL (Erzeugung, Löschung, Routing),
- vorhersagbar in Bezug auf Zeit- und Ressourcenverbrauch.

Das Konzept ermöglicht es einem einzelnen Sendeprozess, Daten an einen oder mehrere Empfangsprozesse zu senden. Ein Empfangsprozess kann auf demselben Verarbeitungselement vorhanden sein, demselben CFM oder sogar einem anderen CFM als dem des Sendeprozesses. Der Sendeprozess hat keinerlei Kenntnis vom Empfangsprozess; er gibt lediglich bestimmte Daten an einen bestimmten VC aus. Ebenso hat der Empfangsprozess keinerlei Kenntnis vom Sendeprozess; er empfängt lediglich bestimmte Daten von bestimmten VC.

Die Quell- und Zielprozesse, die zwischen diesen Prozessen zu übertragenden Datenelemente und die VCs, über die diese gesendet werden, werden während des Systementwurfs definiert. Während der Laufzeit werden diese Informationen durch die RTBP bereitgestellt. Folglich ist der für eine gegebene Systemkonfiguration verwendete VC-Satz fest und bietet einen Bezug für die Erzeugung von Auditdaten.

Jedem VC muss eine bestimmte Nachricht zugewiesen sein. Daher sind den VCs auch folgende Nachrichteneigenschaften zugeordnet:

- Datendarstellung: Eine Nachricht muss in den Blueprints beschrieben sein,
- Sicherheit (Verschlüsselung, Entschlüsselung): Der VC ist gekennzeichnet oder nicht gekennzeichnet,
- Modularität: Eine Anwendung muss als ein Nachrichtenserver gesehen werden. Während des Anwendungsentwurfs ist nicht bekannt, wie viele Benutzer die Nachricht empfangen sollen.

4.2.1.3 TC

Die von der NII angebotene Basiskommunikationsverbindung ist die Transferverbindung (TC). Der Datentransfer über TCs weist folgende Eigenschaften auf:

- die TC ist unidirektional.
- das OS verwaltet die TC im Hinblick auf Erzeugung, Löschung und Routing.
- eine TC muss entweder von einem oder von vielen VC verwendet werden können
- eine TC unterstützt Streaming-Kommunikation.

4.2.1.4 NC

Für die NC legt der ASAAC-Standard keine Anforderungen fest. Die NC müssen innerhalb der MSL verwaltet werden.

Die Eigenschaften einer NC müssen mittels Blueprintdaten konfiguriert werden. Diese Daten sind implementierungsabhängig und müssen daher für die OSL-Software transparent sein.

4.2.2 Datentransferarten

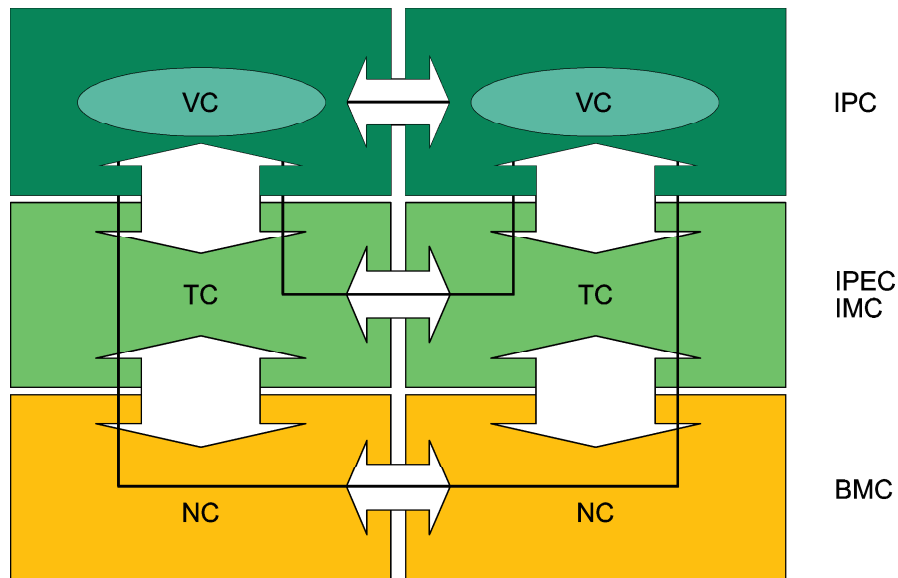


Bild 10 — Datentransferarten

Bild 10 zeigt, ausgehend vom Transferbereich, die vier Kommunikationsarten:

— Intraprozessorkommunikation (IPC)

Kommunikation innerhalb eines Prozessors, die vom VC auf OSL-Ebene gehandhabt wird.

— Intra-PE-Kommunikation (IPEC)

Kommunikation zwischen Prozessoren innerhalb eines Verarbeitungselements, die von der TC auf MSL-Ebene gehandhabt wird.

— Intramodulkommunikation (IMC)

Kommunikation zwischen PE innerhalb eines Moduls, die von der TC auf MSL-Ebene gehandhabt wird.

— Kommunikation zwischen Modulen (BMC)

Kommunikation zwischen CFM, die von einem VC auf OSL-Ebene und von einer TC und einen NC auf MSL-Ebene gehandhabt wird.

4.2.3 Kommunikationskonfiguration

Die ASAAC-Kommunikation muss auf der Grundlage von Blueprint-Daten konfigurierbar sein (siehe 5.3).

Der Geltungsbereich einer lokalen VC-ID ist auf einen einzelnen Anwendungsprozess beschränkt. Der Wert wird durch die Implementierung des zugehörigen Anwendungsprozesses bestimmt. Dieser Wert ist innerhalb dieses Anwendungsprozesses eindeutig. Er ist von der Auslegung der übrigen Anwendungsprozesse unabhängig.

Das GSM verwendet Blueprint-Daten sowohl auf Sende- als auch auf Empfangsseite und muss das OS dazu veranlassen, auf der Basis von Blueprint-Informationen Kommunikationsobjekte zu erzeugen, um die Kommunikation zu konfigurieren:

- das OS erzeugt eine Instanz eines VC. Eine innerhalb des Systems eindeutige globale VC-ID identifiziert einen VC. Sie muss auf Sende- und auf Empfangsseite identisch sein.
- das OS bindet einen anwendungsprozesslokalen VC an einen systemglobalen VC.

- das OS erzeugt eine Instanz einer TC. Eine innerhalb des Systems eindeutige globale TC-ID identifiziert eine TC. Sie muss auf Sende- und auf Empfangsseite identisch sein.
- das OS bildet einen VC auf eine entsprechende TC ab.

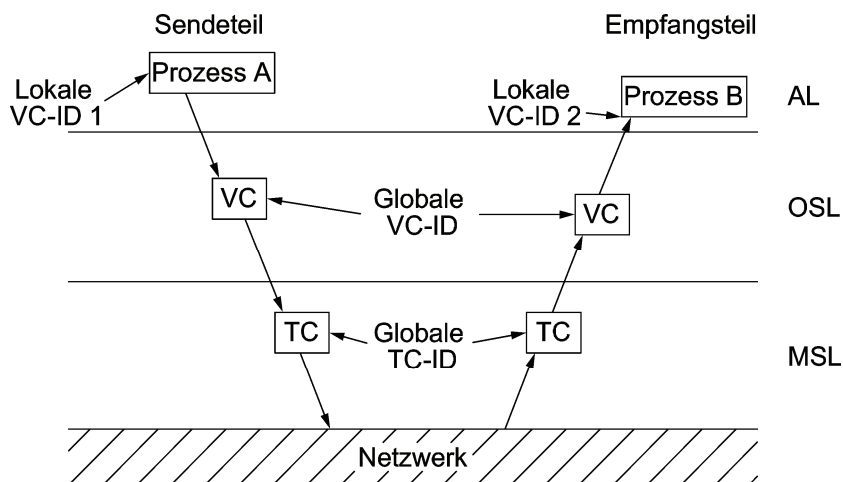


Bild 11 — Kommunikationskonzept

Die Definition der NCs muss Teil der TC-Parametermenge sein, da NCs der Definition einer TC zugeordnet sind. Daher werden NCs nicht als eine separate Entität behandelt.

4.2.4 Kommunikationsprotokolle

4.2.4.1 Einführung

Der ASAAC-Kommunikationsstapel definiert die Protokolle zur Herstellung einer Kommunikation zwischen ASAAC-Schichten.

4.2.4.2 VC zu VC

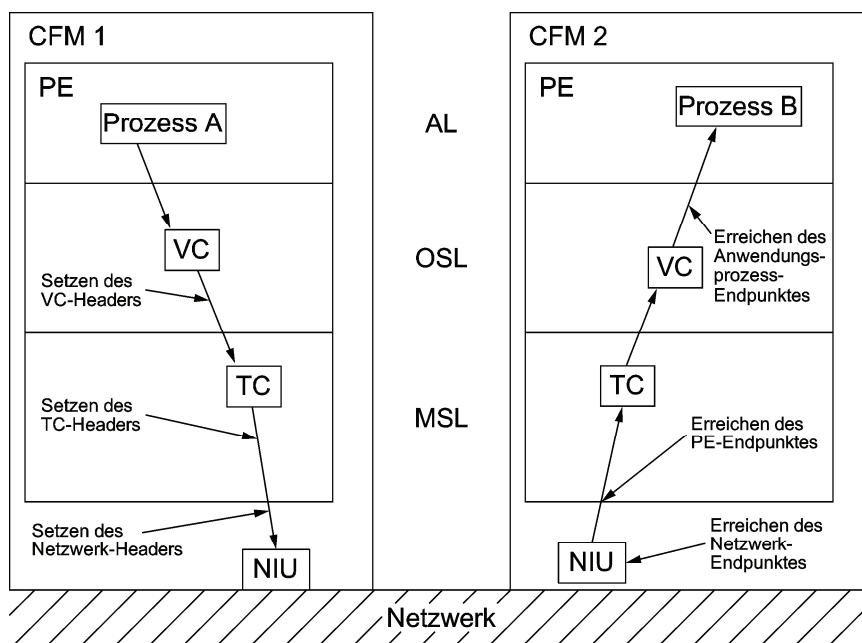


Bild 12 — Routing der Kommunikation zwischen AL

Bei einer Kommunikation zwischen Prozessen auf zwei unterschiedlichen CFM können folgende Mechanismen angewendet werden:

- Prozess-zu-Prozess: Der VC lässt das Erreichen eines Prozesses innerhalb eines PE zu.
- PE-zu-PE: Die TC lässt das Erreichen eines CFM innerhalb eines PE zu.
- CFM-zu-CFM: Der NC lässt das Erreichen eines CFM im Netzwerk zu.

VC-zu-VC unterstützt die Kommunikation zwischen:

- Anwendungsprozessen,
- GSM, die Inter-GSM-Kommunikation ist eine genormte logische Schnittstelle — das GLI, siehe Abschnitt 0,
- Anwendungsmanager und GSM, einer genormten logischen Schnittstelle — der SMLI, siehe 7.3.

4.2.4.3 TC zu TC

Der Zugriff auf die TCs und deren Steuerung muss entweder an der Grenze zwischen OSL und MSL — der NII — erfolgen oder innerhalb der MSL.

Wie oben beschrieben, unterstützt die TC-zu-TC-Kommunikation die BMC-, IMC- und IPE-Kommunikation. Für IPEC und IMC ist der TC-Header erforderlich. Die BMC erfordert sowohl TC- als auch Netzwerk-Header.

Zu beachten ist, dass es sich bei der Inter-MSL-Kommunikation um eine genormte logische Schnittstelle, nämlich die MLI, handelt; siehe 7.4.

4.2.4.4 VC ohne Header-Transfer

Ein VC ohne Header-Transfer ist erforderlich, wenn ein Anwendungsprozess mit einer MSL auf einem anderen PE oder mit Nicht-Kern-Geräten kommuniziert; ein Beispiel dafür ist die Graphikverwaltung, siehe 4.6.

Eine Anwendung sendet einen VC ohne Header:

- der Sendeprozess verwendet einen VC, der auf eine TC abgebildet wird,
- die APOS-VC-Nachricht wird so, wie sie ist, bereitgestellt, ohne dass ein VC-Header angebunden wird,
- ein CFM oder ein Nicht-Kern-Gerät empfängt die TC. Dieser wird je nach Erfordernis entschlüsselt.

Ein VC ohne Header wird empfangen:

- das CFM oder Nicht-Kern-Gerät verschlüsselt die TC-Nutzdaten je nach Erfordernis,
- ein VC ohne Header wird mittels seiner Eigenschaften identifiziert, die durch die Blueprint-VC-Informationen definiert sind,
- erkennt das OS den VC ohne Header als einen solchen, ist er ohne Entfernen des VC-Headers zu verarbeiten,
- die Verwendung eines VC ohne Header bedeutet, dass jede TC, die einen VC ohne Header trägt, auf einen einzelnen VC beschränkt ist.

4.2.4.5 Routing-Header

Um eine Nachricht zwischen Anfangs- und Endpunkt zu leiten, sind folgende Informationen erforderlich:

- ein Header, der Informationen zu seinen Merkmalen enthält,
- ein Bezeichner, der jede Nachrichtenschicht beschreibt. Dieser Bezeichner erlaubt die Identifizierung der Routing-Informationen in internen Tabellen.

Jeder Header muss Informationen zum Erreichen eines Endpunktes bereitstellen (siehe Bild 13):

- der Netzwerk-Header stellt Informationen zum Erreichen des vorgesehenen CFM bereit,
- da der NC von mehreren TCs verwendet werden kann, muss der TC-Header die TC identifizieren. Diese Identifizierung ermöglicht, dass das vorgesehene PE erreicht wird,
- da die TC von mehreren VCs verwendet werden kann, muss der VC-Header den VC identifizieren. Diese Identifizierung ermöglicht, dass der vorgesehene Prozess erreicht wird.

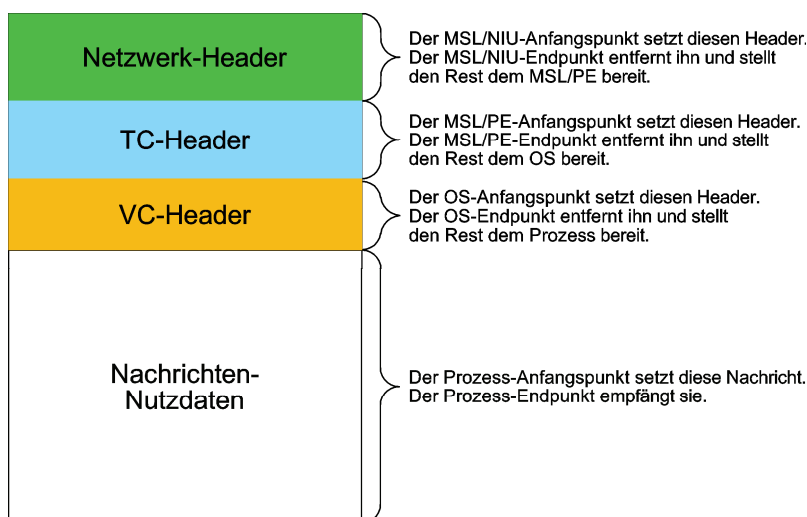


Bild 13 — ASAAC-Nachricht bei BMC-Datentransfer

Die Schicht innerhalb des ASAAC-Kommunikationsstapels, auf der der Transfer erfolgt, bestimmt den Satz der erforderlichen Routing-Daten. Wenn zum Beispiel nur VC-Kommunikationsfunktionen beteiligt sind, ist für IPC lediglich ein VC-Header ausreichend.

Der TC-Header und der VC-Header sind genormt. Tabelle 3 definiert ihre Anwendung auf die Datentransferarten. Der Netzwerk-Header ist nicht genormt, da er technologieabhängig ist.

Tabelle 3 — Routing-Informationen und Datentransfer

Daten-transfer	Netzwerk	TC	VC
BMC	Ja	Ja	Ja
IMC	Nein	Ja	Ja
IPEC	Nein	Ja	Ja
IPC	Nein	Nein	Ja

4.2.5 Multicast

Die ASAAC-Kommunikation muss eine Multicast-Kommunikation unterstützen. Diese Fähigkeit ermöglicht die Verbindung eines Senders mit vielen Empfängern.

Die Multicast-Fähigkeit muss auf VC- und TC-Ebene des ASAAC-Kommunikationsstapels unterstützt werden.

Bei mehreren Empfängern kommt es wahrscheinlich zu einer Kombination von mehreren gleichzeitig angewendeten Datentransferarten (IPC, IPEC, IMC und BMC).

VC-Multicast besitzt einen eindeutigen Bezeichner. Der Senderprozess hat keine Kenntnis vom Multicast-Verhalten. Bei IPC muss dieser VC an alle Empfängerprozesse gebunden sein.

Wenn der VC TCs nutzt (IPEC, IMC und BMC), bestehen zwei Möglichkeiten:

- Verwendung einer einzelnen TC, die Multicast-Fähigkeit bietet,
- Verwendung mehrerer einfacher TCs.

4.2.5.1 TC mit Multicast-Fähigkeit

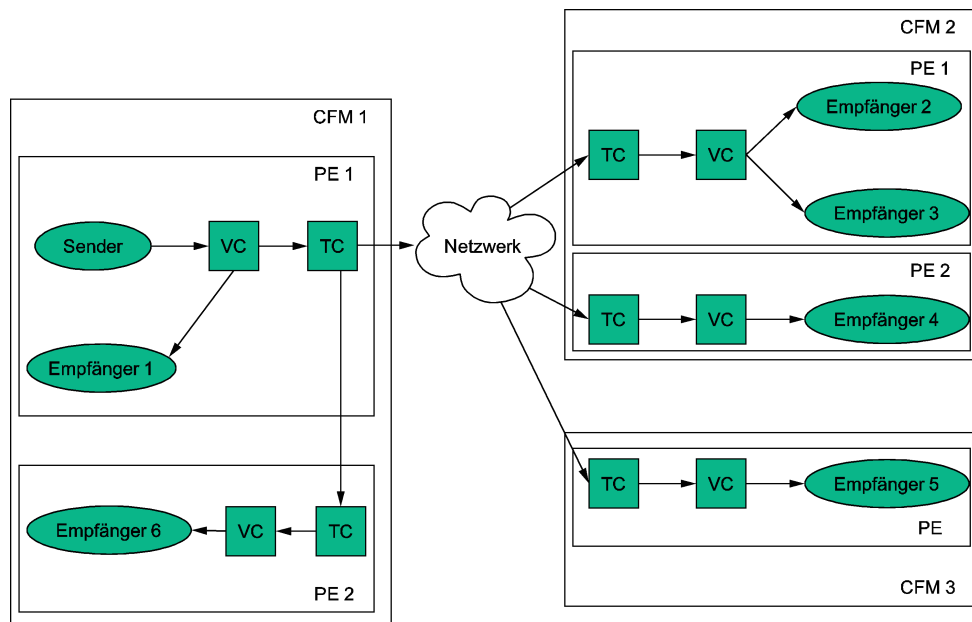


Bild 14 — Multicast-Schema mit einer einzelnen TC

In diesem Beispiel ist der TC-Bezeichner eindeutig. Das Bild zeigt folgende Arten des Datentransfers (vom Sender an):

- Empfänger 1: IPC; der VC ist an den Sender als abgehend und an den Empfänger als eingehend gebunden,
- Empfänger 2 und Empfänger 3: BMC; die eingehende TC ist an einen VC gebunden, dieser eingehende VC ist an beide Empfänger gebunden,
- Empfänger 4: BMC; zu beachten ist, dass die TC auf CFM2/PE1 und CFM2/PE2 denselben Bezeichner hat. Das bedeutet, dass die MSL in der Lage sein muss, mehrere Endpunkte mit demselben Bezeichner zu handhaben,

- Empfänger 5: BMC,
- Empfänger 6: IMC; die abgehende TC von PE1 führt zu Netzwerk und zu PE2. Auch hier wird ein TC mit mehreren Endpunkten gehandhabt.

Für das Durchqueren des Netzwerkes gibt es tatsächlich zwei Verfahren.

1. Die MSL auf NIU-Ebene dupliziert die TC-Nachricht in mehreren Nachrichten, wobei der Netzwerk-Header dem Endpunkt im Netzwerk entspricht.
2. Die MSL auf NIU-Ebene sendet eine eindeutige TC-Nachricht und einen Netzwerk-Header, der das Multicast-Verhalten angibt. Anschließend handhabt die Netzwerkressource den Multicast-Vorgang und führt die Nachricht den Endpunkten zu.

4.2.5.2 Mehrere einfache TCs

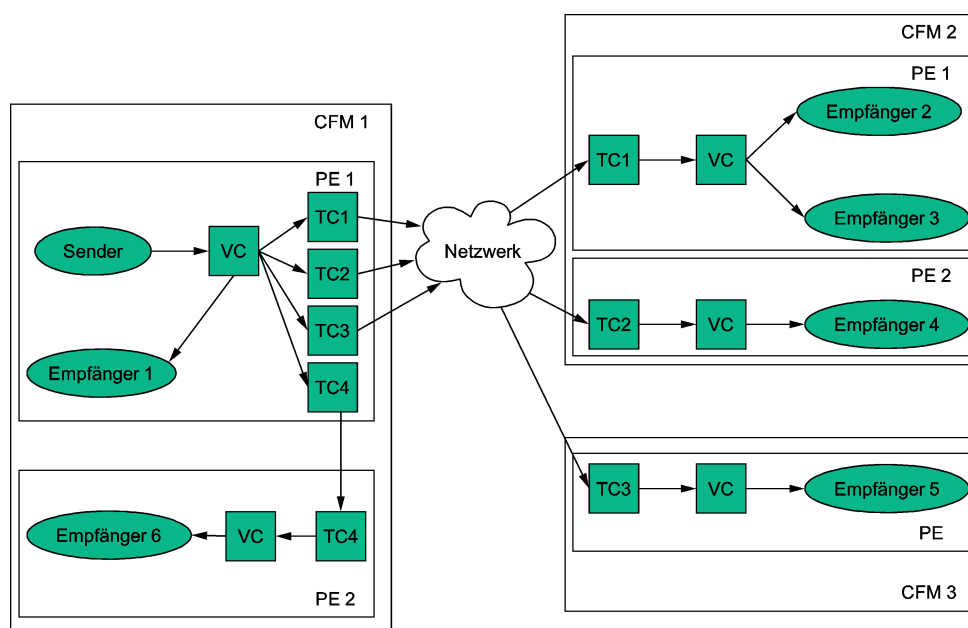


Bild 15 — Multicast-Schema mit mehreren einfachen TCs

Auf Sendeseite ist der Multicast-VC an mehrere TCs gebunden. Auf OS-Ebene wird die VC-Nachricht über mehrere TCs ausgesendet.

4.2.6 Verteiltes Multicast

4.2.6.1 Einführung

Die verteilte Multicast-Kommunikation gilt nur für die Signalverarbeitung. Es handelt sich um eine $N:M$ -Kommunikation (N Sender an M Empfänger), die Multicasting und Fragmentierung nutzt.

Verteiltes Multicast wird verwendet, wenn es erforderlich ist, zur Verringerung der Zeitverarbeitung einer Folge von Signalverarbeitungsoperationen Datenparallelität zu nutzen. Die Eingangsdaten werden in Unterteile aufgeteilt, die von mehreren Prozessoren in derselben Zeit verarbeitet werden. Daten werden von unterschiedlichen Prozessoren verarbeitet, der ausgeführte Algorithmus ist jedoch derselbe.

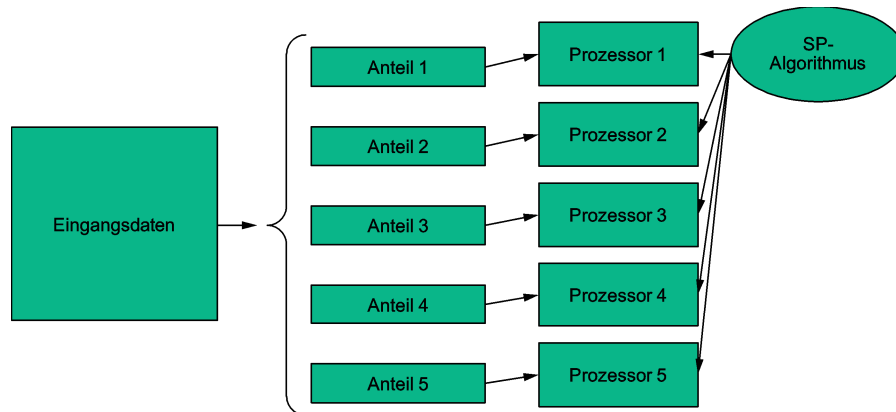


Bild 16 — Datenparallelität

Das verteilte Multicast erlaubt die Neuverteilung von Daten, die über mehrere Prozessoren verteilt sind. Ein typischer Fall für die Anwendung von verteiltem Multicast ist der „Corner-Turn“. Diese Funktion kehrt die Zeilen und Spalten einer Tabelle um, wie unten dargestellt.

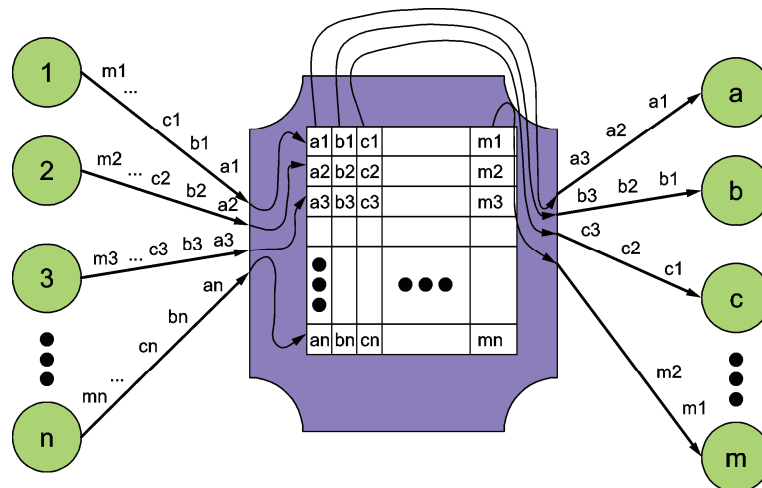


Bild 17 — Corner-Turn

Im oben dargestellten Bild sind $[1 \dots n]$ Sender und $[a \dots m]$ Empfänger. Die umzukehrende Matrix wird zur Laufzeit virtuell vom Kommunikationsmechanismus konstruiert.

Jeder Sender muss für die Adressierung von Daten an jeden Empfänger eine Verteilungsregel anwenden. Eine Verteilungsregel ist ein lineares Gesetz, das definiert, wie ein gesendeter Puffer zu fragmentieren ist, um die einzelnen an jeden Empfänger zu sendenden Puffer zu generieren.

Jeder Empfänger muss für das Adressieren von Daten von jedem Sender eine Sammlungsregel anwenden. Eine Sammlungsregel ist ein lineares Gesetz, das definiert, wie aus den von mehreren Sendern empfangenen fragmentierten Puffern ein Puffer zu bilden ist.

Die $N:M$ -Kommunikation bzw. verteilte Multicast-Kommunikation kann wie folgt betrachtet werden:

- vom Sendeteil aus als eine $1:M$ -Multicast-Kommunikation, bei der die M Empfänger jeweils ein anderes Fragment erhalten,
- vom Empfangsteil aus als eine $N:1$ -Kommunikation, bei der N Sender ein Fragment abgeben.

Der Corner-Turn darf in einem Kontext angewendet werden, der mehrere Dimensionen erfordert. Daten in einer Radaranwendung sind in mehrdimensionalen Arrays organisiert. Im folgenden Beispiel einer Radaranwendung ist ein Wert an einen bestimmten Antennenstrahl, einen bestimmten Impuls und ein bestimmtes Entfernungsfenster (range gate) gebunden. Dies führt unter Verwendung einer Ecke in drei Dimensionen (Impuls, Entfernungsfenster, Strahl).

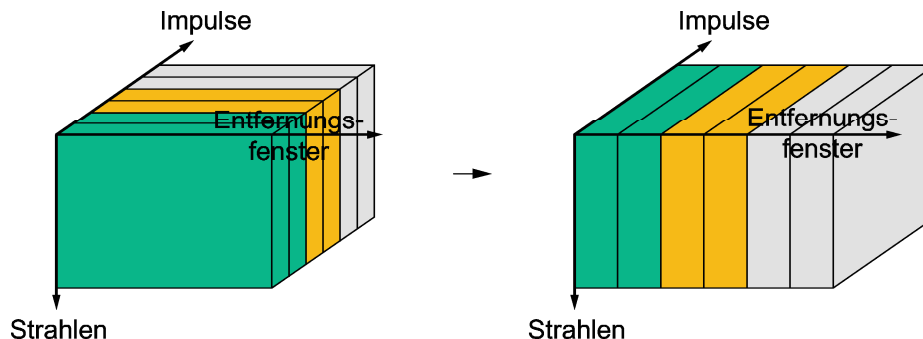


Bild 18 — Corner-Turn in drei Dimensionen

Die einzelnen Scheiben stellen einen Teil des Würfels dar, der von einem Prozessor verarbeitet wird. Sie können überlappen.

4.2.6.2 Anforderungen

Das verteilte Multicast muss folgende Merkmale bieten:

- ein am verteilten Multicast beteiligter Prozessor darf ein Sender und ein Empfänger für diese Transaktion sein,
- jeder Sender und jeder Empfänger darf in einem oder mehreren Verarbeitungselementen angeordnet sein, die wiederum in einem oder mehreren SPM angeordnet sein dürfen,
- für jeden sendenden Prozessor muss für jeden empfangenden Prozessor eine Verteilungsregel definiert werden,
- für jeden empfangenden Prozessor muss für jeden sendenden Prozessor eine Sammlungsregel definiert werden,
- die Signalverarbeitungsanwendung muss in der Lage sein, eine Fragmentierung in einer, zwei oder drei Dimensionen zu verarbeiten,
- jeder Nachricht, die auf einer für die verteilte Multicast-Kommunikation bestimmten TC gesendet wird, müssen der TC-Bezeichner für das Routing an den Empfänger sowie ein Fragmentbezeichner zur Wiederherstellung vorausgehen,

4.2.6.3 Verteilte Multicast-Kommunikation im TLS

Der vorliegende Abschnitt stellt aus der Sicht eines Prozessors vor, wie Daten gesendet und dann empfangen werden.

Die Anwendung muss für das Senden der Daten einen VC verwenden. Der VC ist als ein VC ohne Header festzulegen. Dieser VC ist an eine als DMC-TC festgelegte TC gebunden.

Das OS muss den MOS-Dienst ausführen, der für DMC-Zwecke bestimmt ist: sendFragmentedTransfer.

Die Verteilung der DMC-TC erfolgt innerhalb der MSL. Die einzelnen Fragmente müssen separaten TCs zugewiesen werden. Jedem Empfänger ist eine TC zuzuweisen. Der TC-Bezeichner identifiziert den Empfänger.

Die empfangende MSL muss die TCs mit dem als DMC-TC festgelegten TC-Bezeichner sammeln und die Nachricht auf der Grundlage des Fragment-Bezeichners wiederherstellen, der den Sender identifiziert. Wenn die Wiederherstellung des Puffers abgeschlossen ist, wird er vom MOS-Dienst „receiveFragmentedTransfer“ abgerufen. Der MOS-Dienst darf dieselbe TC-ID verwenden wie der Sendepunkt.

Das OS muss den MOS-Dienst „receiveFragmentedTransfer“ zur Weiterleitung von Daten an die Anwendung durch einen VC ohne Header ausführen.

Bild 19 zeigt ein Beispiel für einen Corner-Turn für die DMC-Verarbeitung auf einem Prozessor.

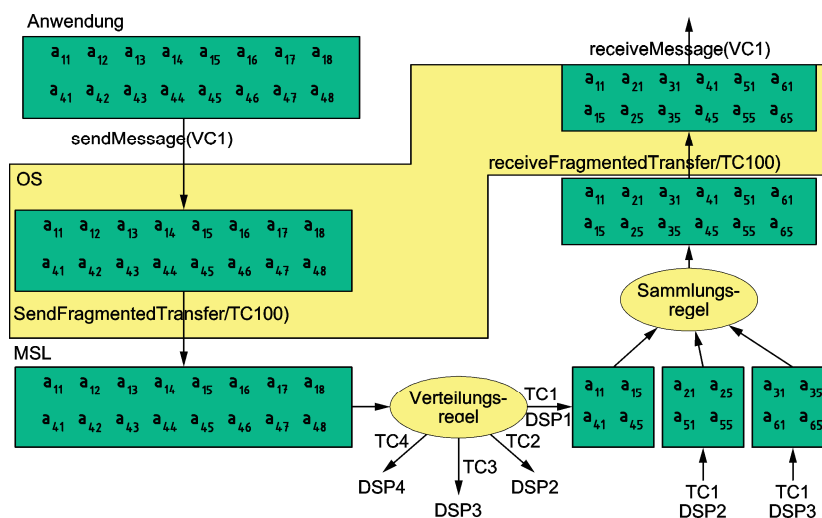


Bild 19 — Darstellung der beteiligten Dienste in DSP1

4.2.6.4 Nachrichtenfragmentierung

Die Datenauswahl der Nachrichtenfragmentierungsprozesse folgt einer linearen Regel, welche, je nachdem, ob diese auf Sende- oder Empfangsseite angewendet wird, entweder eine Verteilungs- oder eine Sammlungsregel bereitstellt.

In dem Corner-Turn-Beispiel müssen Datenadressen nach der folgenden C-ähnlichen Formel berechnet werden:

```
for( a3 = 0; a3 < N3; a3++ ) {
  for( a2 = 0; a2 < N2; a2++ ) {
    for( a1 = 0; a1 < N1; a1++ ) {
      DataAddress = Address0 + a1*Inc1 + a2*Inc2 + a3*Inc3 ;
    }
  }
}
```

Dabei ist

$Address_0$ die Anfangsadresse des internen Puffers, an der mit der Fragmentierung begonnen wird,

N_x die Datenlänge der Dimension X ,

a_x	der aktuelle Index des Dimension X ,
Inc_x	die Schrittweitzahl der Dimension X ,
x	Dimension 1 (erste Dimension), 2 (zweite Dimension), 3 (dritte Dimension).

Wird die Fragmentierung im Kontext von zwei Dimensionen verarbeitet, gilt: N_3 und Inc_3 sind gleich null.

4.2.7 Streaming

Streaming-Daten sind kontinuierliche Daten. Zwei grundlegende Anwendungen sind Video und Audio. Streaming-Daten dürfen nur auf TC-Ebene gehandhabt werden. Dieser Datenstrom darf identifizierte Blöcke von Unterdaten enthalten, z. B. dürfen Videodaten einzelne Bilder enthalten.

Um dem System zu ermöglichen, mit diesem Strom Schritt zu halten, müssen Daten von der Netzwerkschnittstelleneinheit (NIU) an die Verarbeitungseinheit (PU) mit derselben Geschwindigkeit übertragen werden wie Daten, die an der Netzwerkschnittstelleneinheit angekommen sind. Ist diese Anforderung nicht erfüllt, wird der Strom unterbrochen, was als ein Fehler zu behandeln ist.

Bei einer Streaming-TC werden der NIU Informationen bereitgestellt, um den direkten Transfer von Daten an die PU zu erlauben. Die Implementierung der MSL, die die NIU und die PU umfasst, ist projektabhängig. Daher müssen die Konfigurationsdaten, die das Streamingverhalten festlegen, implementierungsabhängig sein.

4.2.8 Datendarstellung

Verschiedene ASAAC-Komponenten eines CFM sind in der Lage, Daten an Komponenten entfernter CFM zu senden. Diese Komponenten befinden sich auf MSL-, OSL- oder AL-Ebene und können Mechanismen zur Datendarstellung umfassen.

Die unterschiedlichen Nachrichtenarten, für die eine Datendarstellung erfolgen kann, sind:

- auf MSL-Ebene:
 - TC, die VC-Nachrichten unterstützt;
 - TC, die MLI-Nachrichten unterstützt;
- auf OSL-Ebene:
 - VC, der Anwendungs-, GLI- und SMLI-Nachrichten unterstützt;
 - VC, der OLI-Nachrichten unterstützt;
- auf APOS-Client-Ebene:
 - Anwendungsnachrichten als typisierte Nachrichten;
 - Anwendungsnachrichten als Nachrichten ohne Header.

Die Datendarstellung beeinflusst jede ASAAC-Schicht.

4.2.8.1 MSL-Ebene

Die Datendarstellung ist auf MSL-Ebene nicht konfigurierbar. Das bedeutet, dass alle Teile einer Nachricht, die in den Verantwortungsbereich der MSL fallen, festkodiert sind.

- TC, die VC-Nachrichten unterstützt: Die TC-Header-Struktur ist fest und bekannt, die Nutznachricht wird nicht modifiziert und wird an das OS übergeben.
- TC, die MLI-Nachrichten unterstützt: Die Struktur aller MLI-Nachrichten — des Headers und der Nutznachricht — ist bekannt. Die Nutznachricht kann verarbeitet und im erwarteten Format an das OS übergeben werden. (Wenn also der CFM-Status angefordert wird, muss die Antwort in einem Format erfolgen, die für den Host-Prozessor verständlich ist.) Weitere Einzelheiten siehe MLI in 7.4.

4.2.8.2 OLS-Ebene

4.2.8.2.1 Kurzbeschreibung

Die Datendarstellung darf, wenn angefordert, auf den VC-Header angewendet werden.

Die Datendarstellung darf nicht auf Nachrichten ohne Header angewendet werden.

Die Datendarstellung muss, wenn angefordert, auf typisierte Nachrichten angewendet werden.

Die Datendarstellung darf nicht auf VC zwischen Prozessen auf demselben PE angewendet werden.

Die Datendarstellung muss auf einer Teilmenge der OMG-CDR beruhen. Es wird die Big-Endian-Ausrichtung vorgeschrieben.

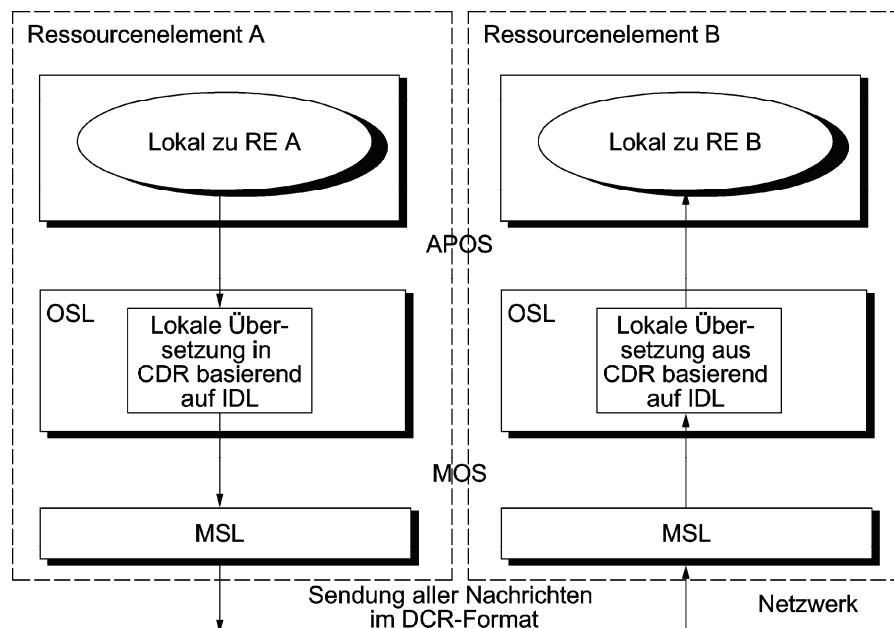


Bild 20 — Datendarstellung

Bei typisierten Nachrichten nimmt das OS gemäß der in Blueprints enthaltenen Informationen eine lokale Übersetzung einer Nachricht vor. Es übersetzt die Nachricht:

- auf Sendeseite in das OMG-CDR-Format. Siehe CORB-Architektur, 15.3,
- auf Empfängerseite in das für den Host-Prozessor verständliche Format.

Bild 20 zeigt die Folge der Übersetzungsaktionen die sich durch eine von Ressourcenelement A gesandte und von Ressourcenelement B empfangene Nachricht ergeben.

4.2.8.2.2 Konfiguration der Datendarstellung

Blueprints enthalten Informationen zur Datendarstellung:

- ein Kennzeichen (Flag) in der SMBP-Struktur „VcToTcMappingDescription“ gibt an, ob der Datendarstellungsmechanismus erforderlich ist. Ist dieses Flag gesetzt, müssen alle VCs auf dieser TC verarbeitet werden, andernfalls darf keine Verarbeitung erfolgen. Bei der Multicast-Kommunikation müssen alle Empfänger dieselben Merkmale aufweisen.
- die SMBP-Struktur „VcDescription“:
 - ein Flag gibt an, ob die Nachricht typisiert oder nicht;
 - wenn die Nachricht als typisiert festgelegt ist, eine anbieterabhängige Darstellung des Nachrichtentyps in der „VcDescription“ im Anschluss an die IDL-Nachrichtendefinition.

Das GSM muss die Verarbeitung der Datendarstellung des OS mit den Blueprint-Informationen konfigurieren.

4.2.8.2.3 OLI-Nachrichten

OLI-Nachrichten sind typisierte Nachrichten. Ihre Beschreibung ist im IDL-Format festgelegt. Da sie genormt sind, sind sie in den Blueprints nicht beschrieben.

Weitere Einzelheiten siehe 7.1.

4.2.8.3 APOS-Clients

Die möglichen APOS-Clients sind die Prozesse in der AL oder dem GSM. APOS-Clients verwalten keine Datendarstellung und dürfen von derartigen Übersetzungsvorgängen keine Kenntnis haben. APOS-Clients senden über VC typisierte Nachrichten oder nicht typisierte Nachrichten.

4.2.8.3.1 Nicht typisierte Nachrichten

Für Nachrichten ohne Header (Bytestrom) müssen Anwendungsdaten unterbrechungsfrei senden und empfangen. In diesem Fall darf der Datendarstellungsmechanismus nicht angewendet werden.

4.2.8.3.2 Typisierte Nachrichten

Um die Datendarstellung auf typisierte Nachrichten anzuwenden, müssen beim Aufbau von APOS-Clients einige Grenzbedingungen beachtet werden:

- die Definitionen der Nachrichtentypen basieren auf einer Teilmenge der IDL-Typen,
- Kompatibilität der Programmiersprachen und Compiler-Optionen für den Aufbau der APOS-Client-Prozesse und der OS-Komponenten, um die genormte Speicheranordnung erzeugen zu können

4.2.8.3.2.1 Definition typisierter Nachrichten

Die Definition der Nachrichtentypen muss auf einer IDL-Teilmenge basieren und Folgendes umfassen:

- einfache Typen,
- zusammengesetzte Typen.

Die folgende Tabelle legt die Teilmenge primitiver IDL-Typen fest, die von einfachen Typen oder zusammengesetzten Typen verwendet werden.

Tabelle 4 — Primitive IDL Typen

IDL-Typ	Oktett-Anordnung	CDR-Darstellung	
octet	1	Nicht interpretierte 8-Bit-Darstellung	
short unsigned short	2	Oktett	Big-Endian
		0	MSB
		1	LSB
long unsigned long	4	Oktett	Big-Endian
		0	MSB
		1	
		2	
		3	LSB
long long unsigned long long	8	Oktett	Big-Endian
		0	MSB
		1	
		2	
		3	
		4	
		5	
		6	
		7	LSB
float	4	IEEE 754-1985	
		Oktett	Big-Endian
		0	s e1
		1	e2 f1
		2	f2
		3	f3
double	8	IEEE 754-1984	
		Oktett	Big-Endian
		0	s e1
		1	e2 f1
		2	f2
		3	f3
		4	f4
		5	f5
		6	f6
		7	f7

Tabelle 4 (fortgesetzt)

IDL-Typ	Oktett-Anordnung	CDR-Darstellung	
long double	16	IEEE 754-1984	
		Oktett	Big-Endian
		0	s e1
		1	e2
		2	f1
		3	f2
		4	f3
		5	f4
		6	f5
		7	f6
		8	f7
		9	f8
		10	f9
		11	f10
		12	f11
		13	f12
		14	f13
		15	f14

Die folgende Tabelle legt die zusammengesetzten Typen fest. Der APOS-Client muss Typen der Spalte „Zusammengesetzter IDL-Typ“ verwenden. Das OS stellt die Datendarstellung sicher und muss diese entsprechend der Spalte „CDR-Kodierung“ kodieren.

Tabelle 5 — Zusammengesetzte IDL-Typen

Zusammengesetzter IDL-Typ	CDR-Kodierung
struct	Komponenten in Definitionsreihenfolge kodiert. Eine Struktur darf aus primitiven Typen und zusammengesetzten Typen aufgebaut sein.
array	Kodierte Array-Elemente in Folge. Das kodierte Element darf entweder ein primitiver Typ oder ein zusammengesetzter Typ sein. Ein Array muss einem eindeutigen Elementtyp zugeordnet sein.
sequence	Anzahl der Elemente, gefolgt von den kodierten Sequenz-Elementen in Folge. Die Anzahl der Elemente muss ein einfacher vorzeichenloser Integertyp sein. Das kodierte Element darf entweder ein primitiver Typ oder ein zusammengesetzter Typ sein. Eine Sequenz (sequence) muss einem eindeutigen Elementtyp zugeordnet sein.
tring	Länge des Strings, gefolgt von der Folge von Zeichen, die in Einzelbyte-Form kodiert sind, einschließlich eines Null-Zeichens. Die Länge des Strings muss ein einfacher vorzeichenloser Integertyp sein. Ein Leerstring hat die Länge eins. Die Länge des Strings muss ein einfacher Integertyp sein. Die kodierten Elemente müssen ein Oktett sein.
union	Diskriminanz-Tag, gefolgt von der Darstellung des gewählten Mitglieds. Das Diskriminanz-Tag muss eine einfache vorzeichenlose ganze Zahl sein. Das gewählte Mitglied darf entweder ein primitiver Typ oder ein zusammengesetzter Typ sein.

Es ist möglich, mehrere zusammengesetzte IDL-Typen mit fester Größe oder variabler Größe zusammenzuführen, solange sie in einer Struktur gekapselt sind und die Zusatzinformation, die durch die CDR-Kodierung ergänzt wird, ebenfalls ein Element der Struktur ist. In der Datendarstellungsspezifikation der Nachricht muss diese Zusatzinformation ein Schlüsselwort in Abhängigkeit vom jeweiligen zusammengesetzten IDL-Typ sein.

- Sequenz (sequence): Element-Anzahl muss Teil der Struktur sein, festgelegt durch ein Schlüsselwort,
- String (string): Länge des Strings muss Teil der Struktur sein, festgelegt durch ein Schlüsselwort,
- Vereinigung (union): Diskriminanz-Tag muss Teil der Struktur sein, festgelegt durch ein Schlüsselwort.

4.3 Sicherheitsmanagement

Sicherheitsmanagement ist die GSM-Funktion, die für den sicheren Transfer von als zu schützend gekennzeichneten Daten durch das System verantwortlich ist. Die Hauptaufgaben des Sicherheitsmanagements sind:

- die Verwaltung und Verteilung von Schlüsseln zur Authentisierung und Verschlüsselung,
- die Bereitstellung von Klassifizierungs- und Freigabedaten für die Referenz-Überwachungsfunktion,
- die Implementierung von Authentisierungs- und Softwareverschlüsselungsalgorithmen,
- die Erstellung eines Sicherheitsauditprotokolls – Eintragung und Mischen/Einfügen.

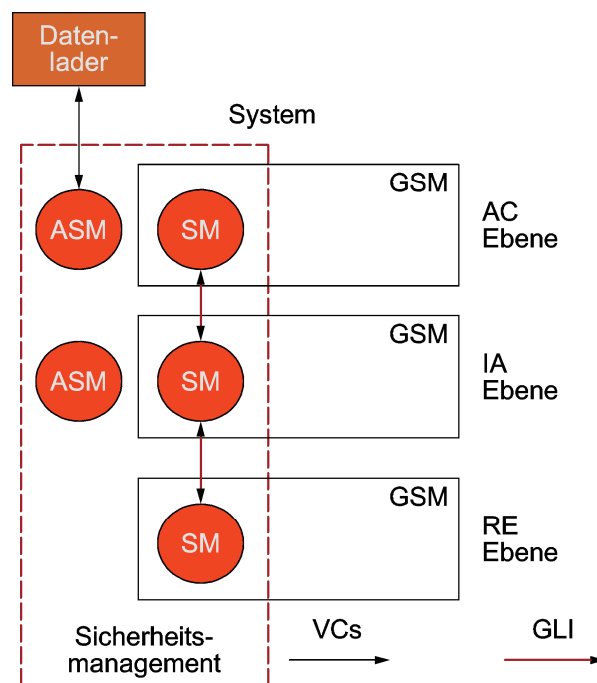


Bild 21 — GSM-Schnittstellen

Die Sicherheitsmanagementfunktion ist gegenwärtig implementiert in Form eines plattformspezifischen Teils, dem Anwendungssicherheitsmanagement (als Teil des AM), und der Hierarchie des Sicherheitsmanagements der generischen Systemverwaltung.

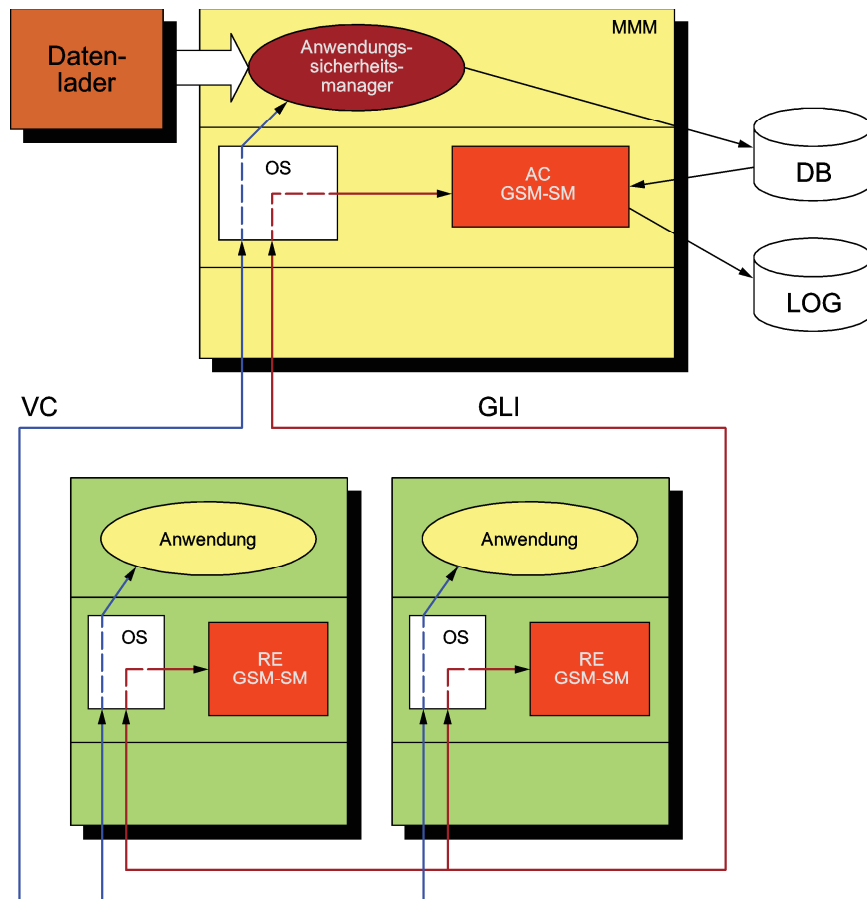


Bild 22 — Wesentliche sicherheitsbezogene Architekturkomponenten

4.3.1 Anwendungssicherheitsmanagement

Das Anwendungssicherheitsmanagement ist der Teil der Sicherheitsmanagementfunktion, der speziell für eine bestimmte Plattform bereitgestellt wird, daher befindet er sich über der APOS neben dem Anwendungsmanager.

Das Anwendungssicherheitsmanagement erfüllt zwei Zwecke:

- es dient als Schnittstelle zwischen dem Datenlader und dem übrigen IMA-System;
- es dient als Schnittstelle zwischen der Datenbank, die die als zu schützend gekennzeichneten Daten hält, und den Funktionsanwendungen, die diese Daten verarbeiten sollen.

ANMERKUNG Der Datenlader und dessen Schnittstelle befinden sich außerhalb des ASAAC-Kerns.

4.3.2 Generisches Sicherheitsmanagement

Das GSM-SM ist der Teil der Sicherheitsmanagementfunktion, der für alle Plattformen gilt.

Das generische Sicherheitsmanagement erfüllt zwei Zwecke:

- es ermöglicht die sichere Übertragung von als zu schützend gekennzeichneten Daten durch Anwendung von Verschlüsselung/Entschlüsselung und Authentisierung;
- es schreibt erkannte Versuche der Sicherheitsverletzung in das Sicherheitsprotokoll.

4.3.3 Verschlüsselung/Entschlüsselung und Authentisierung

Das OS fängt übertragene oder empfangene Daten ab und ermittelt mit Hilfe einer Verweistabelle, ob die übertragenen Daten als zu schützend gekennzeichnet sind. Ist dies der Fall, ist das OS dafür verantwortlich, diese Daten an das GSM-SM zu senden, wobei es die Daten und die VC-ID weitergibt, sodass auf der Grundlage von Blueprint-Informationen an den Daten entsprechende Handlungen vorgenommen werden können, z. B. Verschlüsselung, Entschlüsselung und Authentisierung.

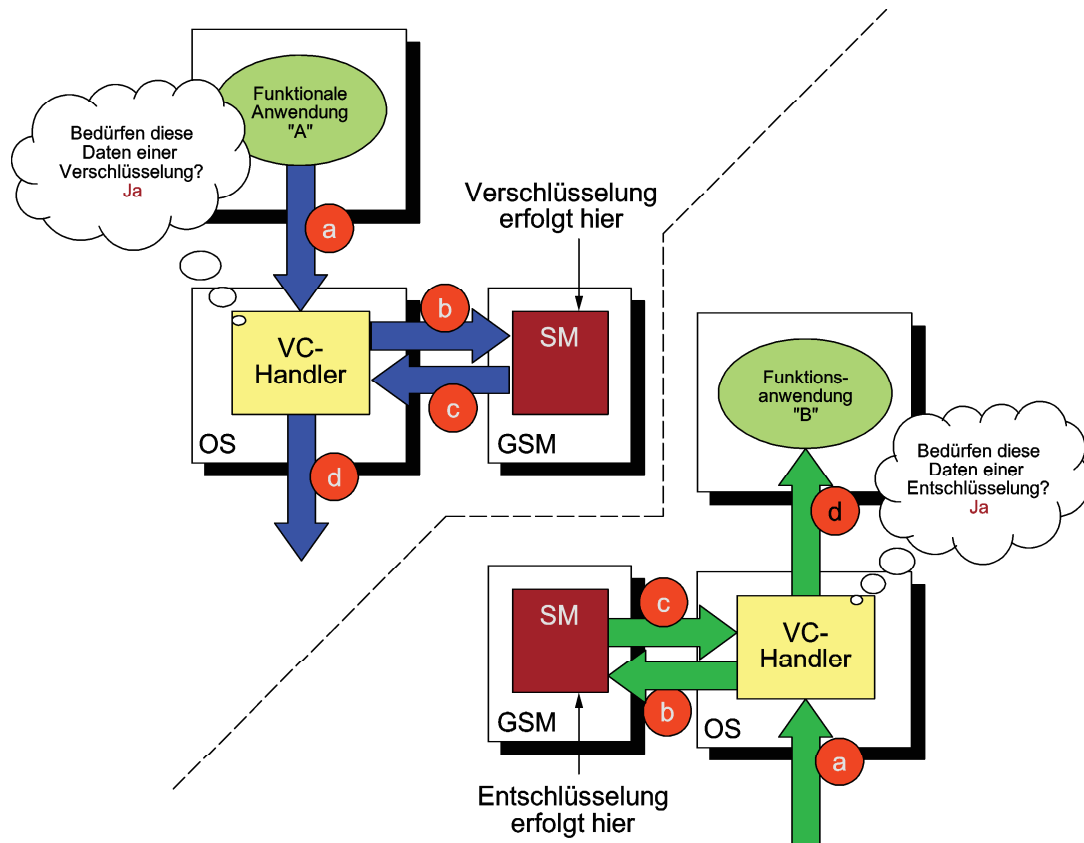


Bild 23 — VC überträgt Daten, die einer Ver- und Entschlüsselung bedürfen

Der Satz der als zu schützend gekennzeichneten Nachrichten hängt vom Systemmodus ab und ist mittels Blueprints definiert. Als Teil des Initialisierungs-/Konfigurationsprozesses führt das GSM-SM, das auf RES resident ist, welche als zu schützend gekennzeichnete Daten verarbeiten, eine Menge vordefinierter Aktionen aus. Die abschließende Aktion des GSM-SM in diesem Betriebsänderungsprozess muss darin bestehen, den SMOS-Dienst „getPMDData“ aufzurufen.

Dabei muss es sich um einen blockierenden Aufruf handeln, der nur dann behandelt wird, wenn das OS die Ausführung einer sicherheitsbezogenen Aktion auf als zu schützend gekennzeichneten Daten erfordert. Dieser Aufruf gibt die ID des VC zurück, der die betreffenden Daten überträgt, sowie die übertragenen Daten. Das GSM-SM verwendet dann die VC-ID als Eintrag in eine Verweistabelle, die während der Konfigurationsphase erzeugt wurde, um zu bestimmen, welche Aktion auf den Daten auszuführen und welcher Schlüssel zu verwenden ist.

Die verarbeiteten Daten werden daraufhin mit Hilfe des SMOS-Dienstes „returnPMDData“ an die OS-VC-Dienste zurückgegeben.

4.3.4 Sicherheitsaudit

Das Sicherheitsaudit erfüllt zwei Zwecke:

- Verlaufsaufzeichnung aller sicherheitsbezogenen Funktionen, um eine Akkreditierung zu unterstützen (auswählbar);
- Protokollierung von versuchten Sicherheitsverletzungen.

Hauptzweck des Sicherheitsaudits ist die Protokollierung von versuchten Sicherheitsverletzungen während des Betriebs des Luftfahrzeugs.

Diese sind dem GSM-SM mit Hilfe des SMOS-Dienstes „getAuditData“ zu melden.

- Das OS muss für die Meldung von Verletzungen der Sicherheitsvorgaben verantwortlich sein.
- Die Referenzüberwachung muss für die Erfassung von Verletzungen der Mehrebenen-Sicherheitsvorgabe verantwortlich sein.
- Das GSM-SM selbst ist dafür verantwortlich, ein Fehlschlagen der Authentisierung zu erkennen.

4.3.5 Sicherheitsreferenzüberwachung

Die Rolle der Sicherheitsreferenzüberwachung ist es, die Implementierung der Sicherheitsvorgaben durchzusetzen und versuchte Verletzungen der Vorgaben zu erkennen. Die Hauptaufgaben der Referenzüberwachung sind:

- die Implementierung eines Algorithmus zur Laufzeit-Freigabe/Klassifizierungsvalidierung (z. B. Bell-LaPadula), d. h. Überprüfen (Checking),
- welcher Prozess-Thread welchen Dienst aufruft,
- die zwischen dem Prozess-Thread und dem Dienst übertragenen Daten.

4.4 Modulverwaltung

Die Modulverwaltung kann in zwei Teile unterteilt sein:

- aus interner Sicht: der Teil, bei dem das CFM hochgefahren wird und dann in einen Zustand übergeht, in dem es bereit ist, den gesamten TLS herunterzuladen,
- aus externer Sicht: der Teil, bei dem ein übergeordnetes CFM die Initialisierung eines untergeordneten CFM steuert.

Die logische Modulschnittstelle (MLI) unterstützt diese beiden Aspekte der Modulverwaltung. Die MLI muss den Zugriff auf ein CFM erlauben, nachdem dieses eingeschaltet und über eine definierte Netzwerkroute initialisiert wurde. Sie muss folgende Funktionen aufweisen:

- Bereitstellung des CFM-PBIT (Selbsttest bei Einschalten),
- Abrufen des aktuellen CFM-Status,
- Abrufen der CFM-Merkmale,
- Herunterladen des ASAAC-Softwarestapels auf ein CFM,
- Konfigurieren eines CFM (für NSM und PCM),
- Synchronisieren der CFM-Zeit,

— Senden und Empfangen von Prüfnachrichten an ein und von einem CFM.

Das MLI-Protokoll muss für alle CFM-Typen gleich sein. Die MLI erlaubt die Initialisierung, Konfigurierung und Synchronisierung der CFM in einem ASAAC-Kernsystem.

Die MLI-Dienste sind in 7.4 festgelegt.

4.5 Massenspeicherverwaltung

4.5.1 Überblick

Das MMM stellt eine Massenspeicherfähigkeit bereit. Es stellt den übrigen CFM die Dateizugriffs- und Speicherfunktionen in einem ASAAC-System bereit.

Das MMM kann, je nach Implementierung, mehrere Dateiarnten enthalten, z. B.:

- Abbilder, die die Komponenten des TLS enthalten: MSL, OS, GSM und Blueprints,
- Software-Anwendungen,
- Datenkonfiguration für das Netzwerk,
- Datenkonfiguration für die Leistungsschalter,
- Datenkonfiguration für die CFM-Kommunikation zum Zeitpunkt des Einschaltens (Routingtabelle),
- Daten für die Anwendung,
- sicherheitsbezogene Daten (z. B. Verschlüsselungsschlüssel).

Das MMM stellt für CFM, die auf diese Dateien zugreifen müssen, Server bereit. Welche Schicht den Server enthält, hängt davon ab, welche Schicht den Requester enthält.

Der direkte Dateizugriff erfolgt nur auf dem MMM. Die übrigen CFM nutzen die OLI zum Fernzugriff auf die Datei.

Zu beachten ist, dass die Massenspeicherverwaltung keine Dateizugriffsrechte für entfernte CFM unterstützt. Dies ist Teil der Sicherheitsverwaltungsvorgaben. Die Grundlage bildet eine sichere Kommunikation und mögliche Verschlüsselung.

4.5.2 Lokale Dateiverwaltung des MMM

Die den Dateizugriff betreffenden Dienste sind spezifisch für das MMM. Außer auf dem MMM darf auf einem CFM keine lokale Dateiverwaltung stattfinden.

In der APOS müssen für das MMM spezifische Dateihandhabungsdienste vorhanden sein. In der MOS müssen für das MMM spezifische Datenspeicherdienste vorliegen.

Die APOS-Dateihandhabung stellt Dienste bereit, die einen Mehrfachzugriff ermöglichen. Das bedeutet, dass mehrere Benutzer in der Lage sein müssen, eine Datei zu verarbeiten (unterschiedliche Prozesse bzw. unterschiedliche Threads innerhalb eines Prozesses). Die unterschiedlichen Benutzer müssen in der Lage sein, gleichzeitige Operationen auszuführen. Nur die Dateizugriffsrechte dürfen Zugriffsbeschränkungen auferlegen.

4.5.3 Zugriff auf Anwendungsdateien

Die Anwendung auf einem entfernten CFM muss durch Kommunikation mit einem MMM auf eine Datei zugreifen. Der anfordernde Anwendungsprozess auf dem entfernten CFM muss als ein Client handeln, und ein Anwendungsprozess auf dem MMM muss als ein Server handeln. Die Kommunikationsverbindung zwischen Client/Server nutzt Kommunikation über VC und basiert auf einem Anfrage/Antwort-Mechanismus. Für die Anfrage ist ein VC erforderlich und für die Antwort ein zweiter VC.

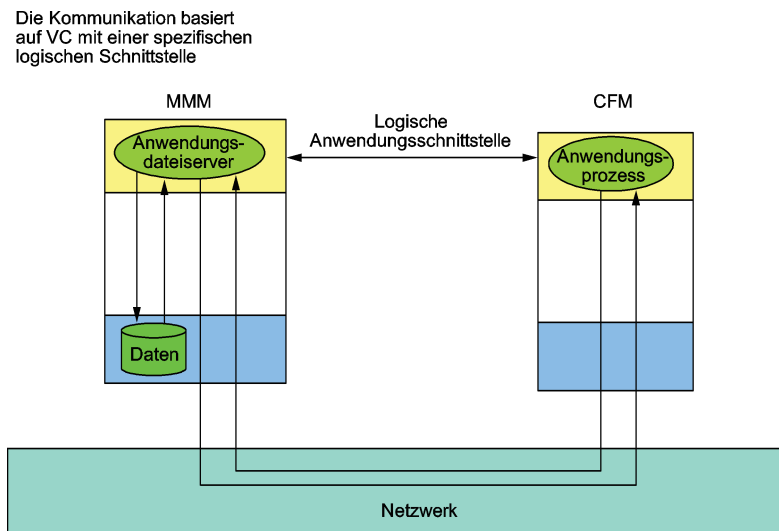


Bild 24 — Dateihandhabung durch eine entfernte Anwendung

Das MMM führt den APOS-Dateihandhabungsdienst aus und sendet das Ergebnis an die anfragende Anwendung.

Hierbei muss es sich um eine spezifische logische Schnittstelle handeln. Ihre Definition hängt von der Implementierung ab.

4.5.4 CFM-Download

Beim Starten verlässt sich ein ASAAC-System darauf, dass ein CFM anfangs eigenständig bootet. Alle anschließend initialisierten weiteren CFM besitzen nur eine residente MSL. Ein CFM mit einem vollständigen Softwarestapel stellt die GSM-Steuerung der Module bereit. Diese Funktion sorgt für das Booten der übrigen CFM.

Software- oder Konfigurationsinformationen können von einem CFM/PE mit einem vollständigen TLS auf ein entferntes PE oder eine entfernte MSU mit nur einer residenten MSL geladen werden.

- Abbilder, die die Komponenten des TLS enthalten: OS, GSM und Blueprints,
- Datenkonfiguration für das Netzwerk,
- Datenkonfiguration für die CFM-Kommunikation zum Zeitpunkt des Einschaltens.

Der Kommunikationsmechanismus dafür ist die logische Modulschnittstelle (MLI) auf der Basis von TCs.

4.5.4.1 Zu bootendes CFM ist ein MMM

Dem MMM ist die Verantwortung für die GSM-Modulsteuerung zugewiesen. Die TCs zwischen dem MMM und einem Satz gesteuerter CFM sind in den Blueprints definiert und wurden konfiguriert.

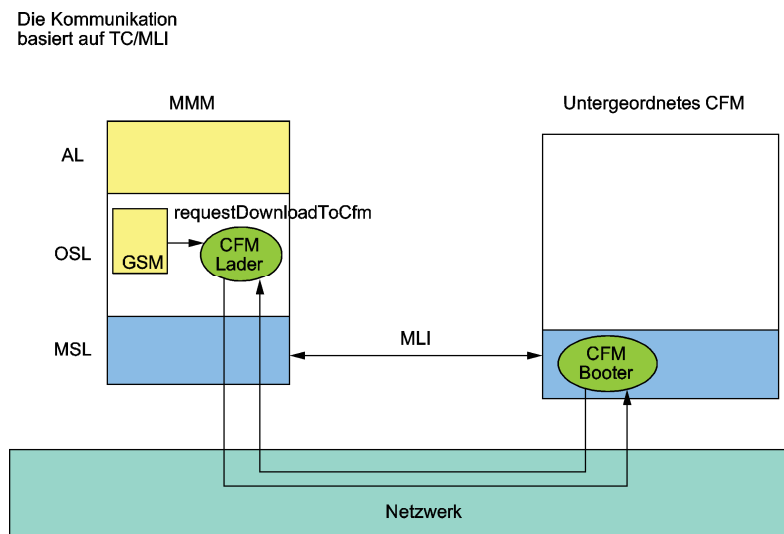


Bild 25 — MMM-Download auf ein CFM, das nur die MSL besitzt

Es wird folgende Download-Sequenz ausgeführt:

- die TCs auf dem MMM sind im Blueprint definiert und wurden konfiguriert,
- beim Lesen des Blueprints initialisiert das GSM einen Download durch einen SMOS-Dienst, bei dem alle notwendigen Informationen an das OS weitergegeben werden,
- anschließend bildet das OS Nachrichten und sendet diese in Ausführung eines dedizierten MOS-Dienstes, der die TC auf der Basis eines mit dem MLI-Downloadmanagement kompatiblen Protokolls nutzt.

Bei der Initialisierung ist auf dem empfangenden CFM eine vorgegebene TC für den Empfang einer MLI-Nachricht bereit.

Zuerst wird die Netzwerkkonfiguration auf das NSM heruntergeladen. Daraufhin sind die übrigen CFM erreichbar.

Anschließend wird die Konfiguration für die CFM-Kommunikation heruntergeladen. Nun ist das CFM in der Lage, auf die MLI-Anforderung zu antworten, da die TCs der antwortenden MLI, die der TC der anfragenden MLI entsprechen, bekannt und konfiguriert sind.

Schließlich können nach der PBIT-Ergebnisanfrage die Abbilder heruntergeladen werden, die die TLS-Komponenten enthalten.

4.5.4.2 Zu bootendes CFM ist kein MMM

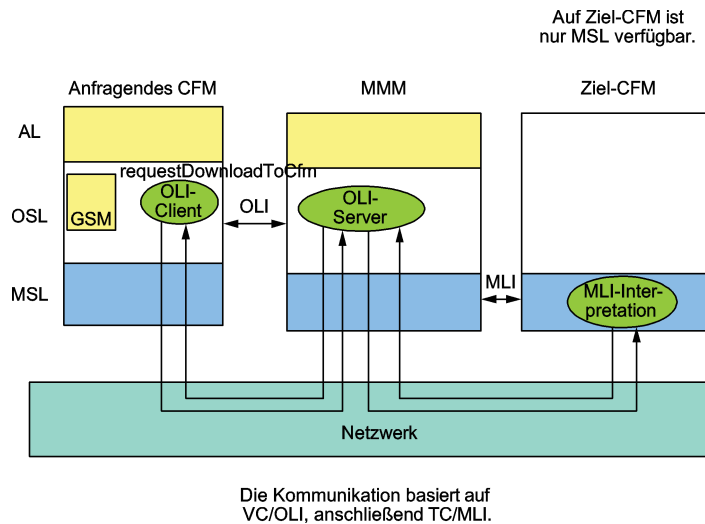


Bild 26 — CFM-Download auf ein CFM, das nur die MSL aufweist

In diesem Fall ist das CFM nicht in der Lage, Dateien eigenständig auf ein Ziel-CFM herunterzuladen.

Es wird folgende Download-Sequenz ausgeführt:

- die VCs zwischen dem MMM und dem anfragenden CFM sind in den Blueprints definiert und wurden konfiguriert,
- die TCs zwischen den MMM und dem Ziel-CFM sind in den Blueprints definiert,
- beim Lesen der Blueprints initialisiert das anfragende CFM/GSM einen Download durch einen SMOS-Dienst, bei dem alle notwendigen Informationen an das OS weitergegeben werden,
- das anfragende CFM/OS bildet Nachrichten und sendet die OLI-Anfragenachricht unter Verwendung des VC auf der Basis eines mit dem OLI-Downloadmanagement kompatiblen Protokolls,
- das MMM/OS empfängt die OLI-Anfrage und greift auf die herunterzuladende Datei zu,
- das MMM/OS führt den Download auf die MLI aus,
- wenn der Downloadprozess beendet ist, sendet das MMM/OS eine OLI-Antwort auf dem VC an das anfragende CFM/OS. Die VC-ID der OLI-Antwort wird über eine Regel berechnet, die von der OS-Implementierung abhängig ist.

Das anfragende CFM kann die drei im ersten Fall genannten Downloads einleiten.