OSEK / COM

Florian Hohnsbehn

PG AutoLab
Seminarwochenende 21.-23. Oktober 2007

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einführung

1.1 Was ist OSEK COM?

OSEK COM ist eine vom OSEK-VDX-Konsortium entwickelte Kommunikationsumgebung für Anwendungssoftware für automotive Steuergeräte. OSEK COM beschreibt die Schnittstellen und das Verhalten für die Kommunikation sowohl innerhalb einer elektronischen Steuereinheit (interne Kommunikation) als auch zwischen vernetzten Knoten im Fahrzeug (externe Kommunikation). Die Spezifikation beschreibt dabei dabei das Verhalten innerhalb einer elektronischen Steuereinheit (ECU). Sie setzt voraus, dass OSEK COM zusammen mit einem OSEK-konformen Betriebssystem verwendet wird. Für andere Betriebssysteme definiert die Spezifikation Anforderungen für den Betrieb von OSEK COM.
Dieser Text bezieht sich auf die zum Zeitpunkt seiner Erstellung aktuellste Version der Spezifikation, die Version 3.0.3 vom 20. Juli 2004.

Ziele und Anforderungen

Das Ziel der Entwicklung von OSEK COM war, eine einheitliche Kommunikationsumgebung für automotive Steuergeräte zu schaffen. Diese sollte einheitliche Schnittstellen für die Kommunikation zwischen Tasks- und/oder Interrupt-Serviceroutinen auf einem oder zwischen verschiedenen Steuergeräten bieten.
Um eine möglichst universell einsetzbare Umgebung zu bieten und damit Kosten und Aufwand einzusparen, hatten bei der Entwicklung Portierbarkeit, Wiederverwendbarkeit und Interoperabilität eine besondere Bedeutung. OSEK COM soll auf verschiedenen Anwendungstypen eingesetzt werden können und unabhängig vom im System verwendeten Kommunikationsprotokoll sowie der zugrunde liegenden Netzwerkstruktur bzw. des zugrunde liegenden Bussystems sein. Dabei soll die Schnittstelle diese Unterschiede sowie die zwischen interner und externer Kommunikation für die Anwendung verstecken.
OSEK-COM-Anwendungen sollen auf Systemen mit unterschiedlicher Hardware laufen. Daher ist es wichtig, dass je nach verfügbaren Ressourcen oder erforderter Funktionalität im jeweiligen System nur minimale Ressourcen benötigt werden. Um diese Skalierbarkeit umzusetzen, wurden verschiedene Funktionalitätsstufen, die sogenannten Conformance Classes, definiert.
Zudem soll das indirekte Netzwerkmanagement, das in OSEK NM definiert ist, unterstützt werden.

1.2 Einordnung in OSEK

Die Kommunikation in OSEK COM schließt die in Abbildung 1 dargestellten drei Schichten ein: Interaction Layer, Network Layer und Data Link Layer. Während für die Network Layer sowie die Data Link Layer lediglich minimale Anforderungen definiert werden, stellt die Interaction Layer die in OSEK COM definierte API für die Anwendung zur Verfügung. Die Data Link Layer bietet den über ihr liegenden Schichten, also der Interaction Layer bzw. der Network Layer, Dienste für den Datentransfer. Dieser Artikel konzentriert sich auf die Interaction Layer und erwähnt nur kurz die Anforderungen an die Network Layer und die Data Link Layer.

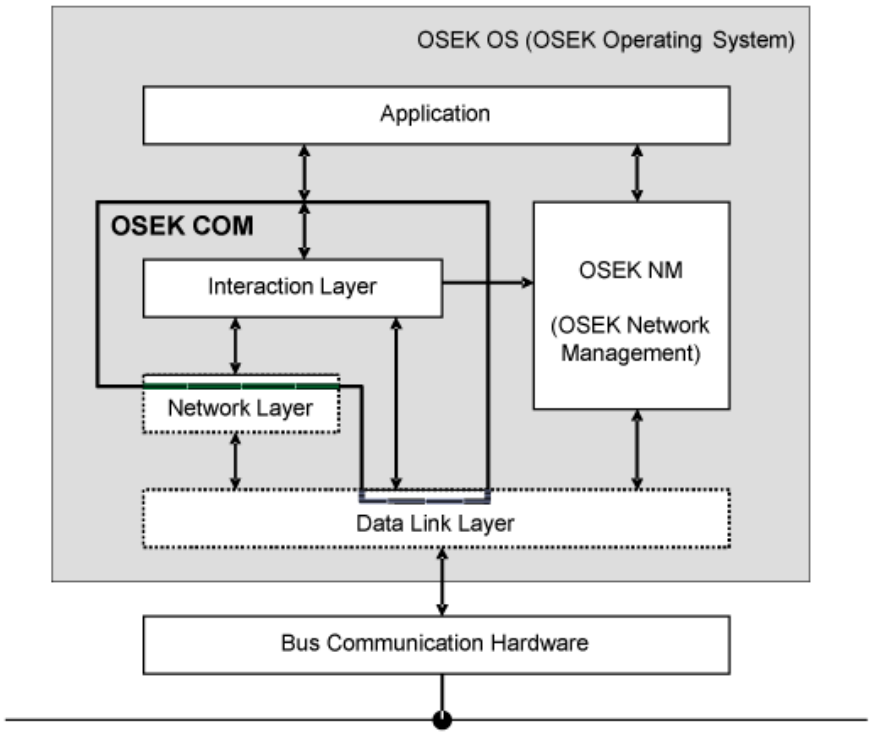

Abbildung 1: Schichtenmodell in OSEK COM

2 Nachrichten

2.1 Kommunikationskonzept

Die Kommunikation in OSEK COM ist nachrichten- oder auch signalbasiert. Der Begriff einer Nachricht beschreibt dabei nicht die Information, die bei der Kommunikation übermittelt wird, sondern vielmehr eine Art Träger für zu übertragene Informationen von der Anwendung zur Interaction Layer und umgekehrt. Nachrichten sowie deren Eigenschaften werden bei der Systemerstellung statisch in der OSEK Implementation Language (OIL) festgelegt und können nicht zur Laufzeit geändert werden. Weiterhin können zur Laufzeit keine Nachrichten hinzugefügt oder entfernt werden. Datenübertragungen können also nur über bei der Systemerstellung definierte Nachrichten erfolgen. Zu den Eigenschaften einer Nachricht gehört die Länge der Nachricht. Diese kann entweder statisch, dynamisch oder 0 sein. Bei Nachrichten der Länge 0 spricht man von Zero-Length-Messages.
In den kommunizierenden Steuereinheiten werden ebenfalls zum Zeitpunkt der Systemerstellung Nachrichtenobjekte zum Senden (Sending Message Object) und Empfangen (Receiving Message Object) definiert. Receiving Message Objects können entweder die Eigenschaft Queued oder Unqueued haben. Bei Nachrichtenobjekten mit der Eigenschaft Unqueued werden die Daten erst von der nächsten übertragenen Nachricht überschrieben. Bis zu diesem Zeitpunkt, kann

sie beliebig oft ausgelesen werden. Bei Nachrichtenobjekten mit der Eigenschaft Queued wird immer die älteste Information der Queue ausgelesen und nach dem Lesen sofort gelöscht. Neu ankommende Nachrichten werden ans Ende der Queue geschrieben. Für jedes Nachrichtenobjekt wird die maximale Länge der Queue festgelegt. Kommt eine Nachricht bei einem empfangenden Nachrichtenobjekt an, und ist dessen Queue bereits voll, geht die ankommende Nachricht verloren.

Interne Kommunikation

Die interne Kommunikation findet komplett innerhalb der Interaction Layer statt. Nachdem die Nachricht vom Sender in der Anwendung zum Sending Message Object übertragen wurde, wird diese unmittelbar dem Receiving Message Object zur Verfügung gestellt.

Externe Kommunikation

Bei der externen Kommunikation benutzt die Interaction Layer Dienste der unteren Schichten. Aus dem sendenden Nachrichtenobjekt wird die Nachricht in eine Interaction Layer Protocol Data Unit (I-PDU) kopiert. Eine I-PDU kann mehrere Nachrichten enthalten. Als nächstes wird bei der unteren Schicht die Übertragung beantragt und bei Bestätigung die Nachrichten an Protocol Data Units der zugrunde liegenden Schicht übertragen. Liegt eine Nachricht für eine Steuereinheit zum Empfang vor, informiert die untere Schicht die Steuereinheit darüber. Die zu empfangende(n) Nachricht(en) werden dann in einem I-PDU, das für den Empfang vorgesehen ist, empfangen. Von dort werden die Nachrichten in das empfangende Nachrichtenobjekt kopiert und anschließend an den Empfänger in der Anwendung übertragen.
Abbildung 2 stellt den Ablauf der internen wie externen Kommunikation vereinfacht da.

2.2 Nachrichtenübermittlung

Zum Senden von Nachrichten stellt OSEK COM der Anwendung die API-Dienste SendMessage, SendDynamicMessage und SendZeroMessage zur Verfügung, mit denen die Übertragung der verschiedenen Nachrichtentypen gestartet werden kann.

Ablauf der Übermittlung

Zunächst wird die Nachricht ans sendende Nachrichtenobjekt geschickt. Bei der internen Kommunikation werden die Daten dann direkt an das empfangende Nachrichtenobjekt geroutet, von wo aus es an den Empfänger geschickt wird.
Bei der externen Kommunikation werden die Nachrichten zunächst gemäß eines je Nachricht festgelegten Algorithmus gefiltert. OSEK COM bietet eine Menge von 15 verschiedenen Filtern, die später kurz vorgestellt werden. Die Filter können nur auf Nachrichten mit statischer Länge angewandt werden. Wird eine Nachricht ausgefiltert, wird sie verworfen und nicht weiter behandelt. Nicht ausgefilterte Nachrichten werden in eine für das Senden verantwortliche I-PDU

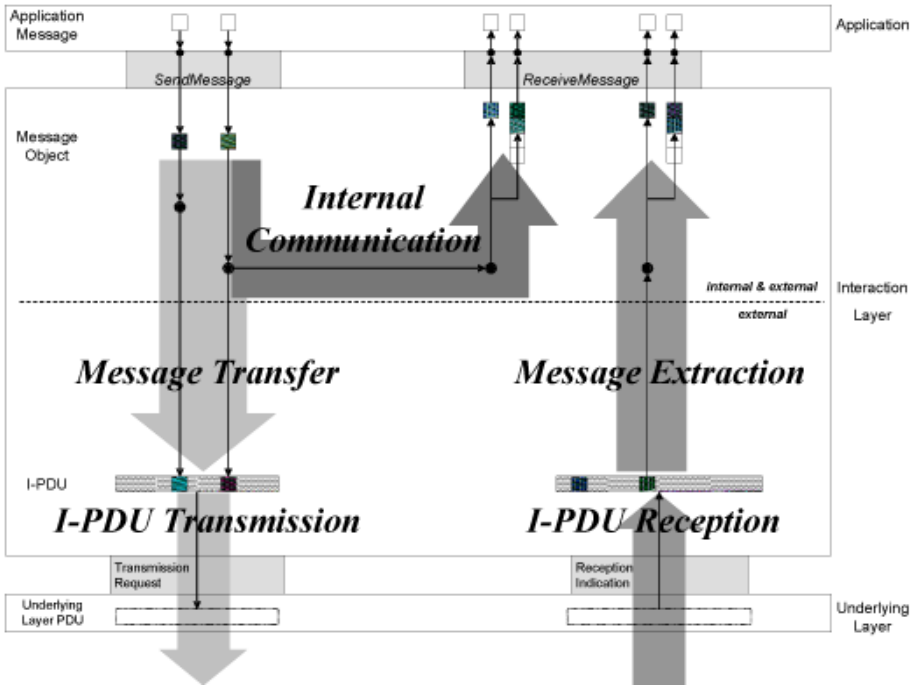

Abbildung 2: Vereinfachtes Modell der Kommunikation

gepackt und an eine PDU der unteren Schicht übertragen. Falls vorgesehen erfolgt eine Bestätigung, ob die Nachricht erfolgreich übertragen wurde.
Bei Zero-Length-Messages werden keine Daten übertragen.

Transfer properties: Triggered vs. Pending

Bei der Übertragung der Nachricht vom sendenden Nachrichtenobjekt zur I-PDU gibt es zwei unterschiedliche Transfereigenschaften: Triggered und Pending. Diese Eigenschaft wird je Nachricht definiert.
Bei Nachrichten, die mit der Eigenschaft Triggered abgeschickt werden, wird direkt, nachdem die Nachricht in die I-PDU geschrieben wurde, die Übertragung an die zugrunde liegende Schicht veranlasst. Nachrichten der Länge 0 haben immer die Eigenschaft Triggered.
Bei Nachrichten mit der Eigenschaft Pending wird die Nachricht in die I-PDU geschrieben, jedoch wird noch keine Übertragung an die untere Schicht veranlasst. Die Übertragung erfolgt erst dann, wenn das nächste Mal eine Nachricht mit der Eigenschaft Triggered die I-PDU erreicht. Um die Übertragung einer Nachricht mit der Eigenschaft Pending zu bewirken, können Zero-Length-Messages eingesetzt werden.

Transmission modes: Direct, Periodic, Mixed

Für die Übertragung der Nachrichten aus der I-PDU des Steuergeräts an eine PDU der darunter liegenden Schicht gibt es drei verschiedene Modi: den direkten Modus, bei dem die Nachricht bei Bedarf, also auf Aufforderung, an die

untere Schicht übertragen wird; den periodischen Modus, bei dem der Inhalt der I-PDU in regelmäßigen Abständen übertragen wird; der dritte Modus ist der gemischte Modus, der eine Kombination aus dem direkten und dem periodischen Modus ist. Standardmäßig wird im direkten Modus übertragen. Durch den API-Dienst StartPeriodic kann während der Laufzeit in den periodischen oder gemischten Modus gewechselt werden.
Im direkten Übertragungsmodus wird jedes Mal, wenn eine Nachricht mit Triggered Property in einer I-PDU ankommt, eine Übertragung der I-PDU ausgelöst. Es kann allerdings eine minimale Verzögerung definiert werden, die zwischen zwei Sendevorgängen verstreichen muss. Der Timer für die Verzögerung startet, sobald eine Übertragung bestätigt wird. Wird nach einer Übertragung eine weitere Übertragung angefordert, bevor die minimale Verzögerunszeit verstrichen ist, wird die zweite Übertragung erst ausgeführt, sobald die minimale Verzögerung abgelaufen ist. Kommen innerhalb der minimalen Verzögerungszeit zwei Nachrichten mit Triggered Property an, wird die erste verworfen und von der zweiten überschrieben. Dies ist in Abbildung 3 dargestellt.
Im periodischen Modus wird die I-PDU mit jedem SendMessage-Aufruf oder

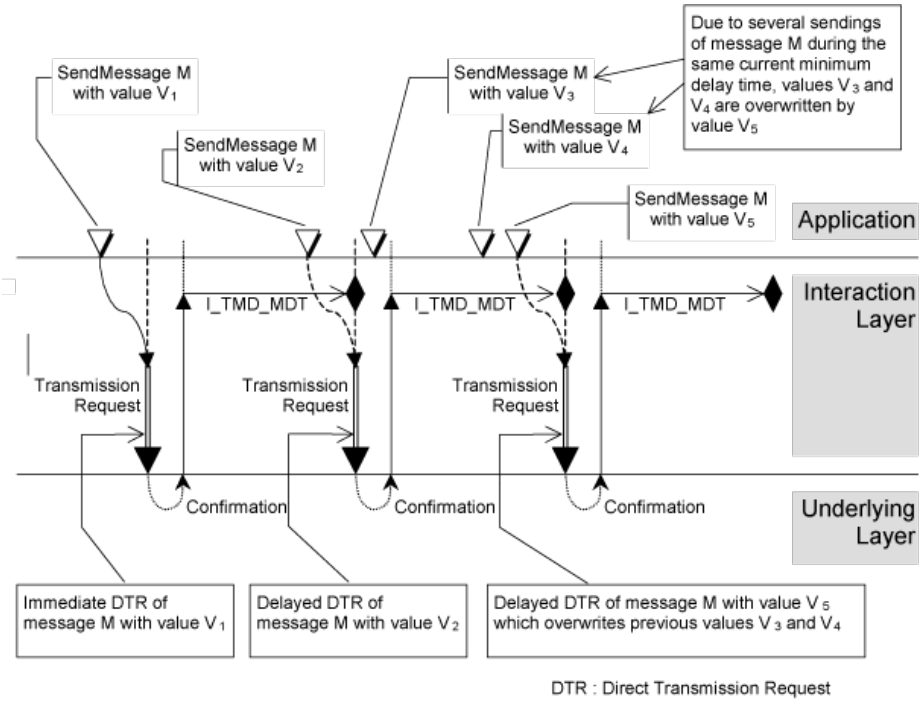

Abbildung 3: Direkter Übertragungsmodus mit Verzögerung

SendDynamicMessage-Aufruf aktualisiert, ohne dass sofort eine Übertragungsforderung gestellt wird. Dabei wird auch die Transfereigenschaft, also insbesondere die Eigenschaft Triggered, ignoriert. Die Übertragung wird immer und ausschließlich periodisch ausgelöst. Die Periode sowie die Zeit bis zur ersten Übertragung (Offset) werden mit dem Aufruf von StartPeriodic festgelegt.
Der gemischte Modus ist wie erwähnt eine Kombination der beiden zuvor genannten Übertragungsmodi. Zum einen werden Übertragungen periodisch durch-

geführt, zum anderen können zusätzlich jederzeit Nachrichten mit der Triggered-Eigenschaft eine direkte Übertragung veranlassen. Wie beim direkten Modus kann auch hier eine minimale Verzögerung definiert werden. Dabei kann die minimale Verzögerung auch eine periodische Übertragung verzögern. Diese wird dann nachgeholt, sobald die Verzögerung abgelaufen ist. Diese Verzögerung kann sich auch noch auf weitere periodische Übertragungen auswirken. Die Periode selbst wird dadurch aber nicht verschoben. Abbildung 4 zeigt ein solches Beispiel.

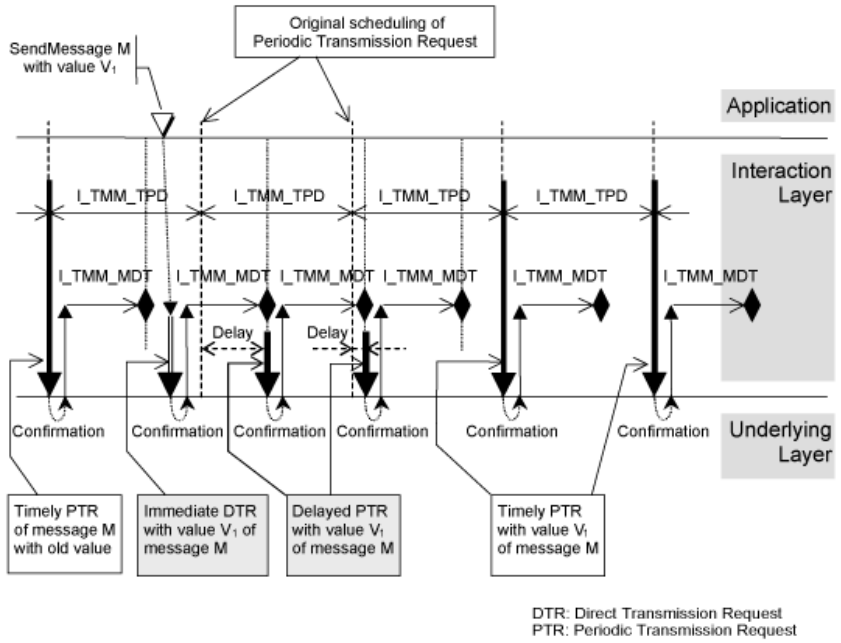

Abbildung 4: Gemischter Modus mit Verzögerung

2.3 Nachrichtenempfang

Beim Empfang einer externen Nachricht teilt die untere Schicht der Steuereinheit mittels einer Indication mit, dass eine Nachricht empfangen werden soll. Falls kein Fehler gemeldet wird, war die Übertragung erfolgreich, und die Daten aus der PDU der unteren Schicht werden in die für den Empfang zuständige I-PDU übertragen. Falls ein Fehler aufgetreten ist, kann dieser gemeldet werden. Nachdem die Nachrichtendaten in die I-PDU übertragen wurden, werden sie aus dieser entpackt. Danach ist der Ablauf für interne wie externe Nachrichten derselbe. Zunächst findet wieder eine Filterung statt. Dabei sind die gleichen Filteralgorithmen wie beim Senden verfügbar. Ausgefilterte Nachrichten werden auch hier verworfen und nicht weiter behandelt. Nicht ausgefilterte Nachrichten werden ins empfangende Nachrichtenobjekt der Steuereinheit kopiert. Nun benachrichtigt die Interaction Layer die Anwendung über die vorliegende Nachricht, und diese wird von der Anwendung empfangen.

2.4 Filter

Wie bereits erwähnt, können sowohl beim Versenden als auch beim Empfangen von Nachrichten Filteralgorithmen benutzt werden. Diese können jedoch nur auf Nachrichten mit statischer Länge angewendet werden. Die einfachsten Filter sind true bzw. false, bei denen alle bzw. gar keine Nachricht durchgelassen werden. Für einige der Filter müssen zusätzliche Attribute gespeichert werden. Um den letzten Wert mit dem neuen Wert der Nachricht vergleichen zu können, müssen die beiden Attribute New value und Old value gespeichert werden. Zudem kann der Wert mit einer konstanten Maske (mask) versehen werden und dann mit dem alten Wert verglichen werden. Ein Vergleich mit einem konstanten Wert oder mit einem Minimum oder Maximum (x, min, max) ist ebenfalls möglich. Um nicht jedes Auftreten einer Nachricht zu betrachten, sondern nur alle n Auftreten, können die konstanten Werte period und offset gesetzt werden und die Anzahl der Auftreten mit dem Zähler occurence mitgezählt werden. Abbildung 5 zeigt alle verfügbaren Filteralgorithmen.

Algorithm Reference	Algorithm	Description
F_Always	True	No filtering is performed so that the message always passes
F_Never	False	The filter removes all messages
F_MaskedNewEqualsX	*(new_value&mask) == x*	Pass messages whose masked value is equal to a specific value
F_MaskedNewDiffersX	*(new_value&mask) != x*	Pass messages whose masked value is not equal to a specific value
F_NewIsEqual	*new_value == old_value*	Pass messages which have not changed
F_NewIsDifferent	*new_value != old_value*	Pass messages which have changed
F_MaskedNewEqualsMaskedOld	*(new_value&mask) == (old_value&mask)*	Pass messages where the masked value has not changed
F_ MaskedNewDiffersMaskedOld	*(new_value&mask) != (old_value&mask)*	Pass messages where the masked value has changed
F_NewIsWithin	*min <= new_value <= max*	Pass a message if its value is within a predefined boundary
F_NewIsOutside	*(min > new_value) OR (new_value > max)*	Pass a message if its value is outside a predefined boundary
F_NewIsGreater	*new_value > old_value*	Pass a message if its value has increased
F_NewIsLessOrEqual	*new_value <= old_value*	Pass a message if its value has not increased
F_NewIsLess	*new_value < old_value*	Pass a message if its value has decreased
F_NewIsGreaterOrEqual	*new_value >= old_value*	Pass a message if its value has not decreased
F_OneEveryN	*occurrence % period == offset*	Pass a message once every N message occurrences. Start: occurrence = 0. Each time the message is received or transmitted, occurrence is incremented by 1 after filtering. Length of occurrence is 8 bit (minimum).

Abbildung 5: Filteralgorithmen

2.5 Deadline Monitoring

Für jede Nachricht kann ein Deadline Monitoring sowohl für den Empfang als auch für den Versand von Nachrichten konfiguriert werden. Dieses wird für jedes I-PDU, das diese Nachricht enthält, ausgeführt.
Beim Senden von Nachrichten wird mit Hilfe des Deadline Monitorings überwacht, ober Übertragungsanfragen auch wirklich Übertragungen zur Folge ha-

ben, also ob innerhalb eines festgelegten Zeitintervalls eine Übertragung erfolgt, also eine Empfangsbestätigung der unteren Schicht folgt. Da im direkten Übertragungsmodus für Nachrichten mit der Eigenschaft Pending keine Übertragung angefordert wird, ist für diese Nachrichten im direkten Modus kein Deadline Monitoring möglich. Wurde der Empfang einer Nachricht nicht innerhalb des Zeitintervalls bestätigt, liegt ein Time-Out vor, und die Anwendung wird darüber informiert. Sie entscheidet dann, ob die Übertragung erneut versucht werden soll. Das Deadline Monitoring funktioniert für jeden Übertragungsmodus ähnlich. Abbildung 6 zeigt ein Beispiel mit Time-Out für das Deadline Monitoring im periodischen Modus. Beim Empfang ist das Deadline Monitoring für jeden

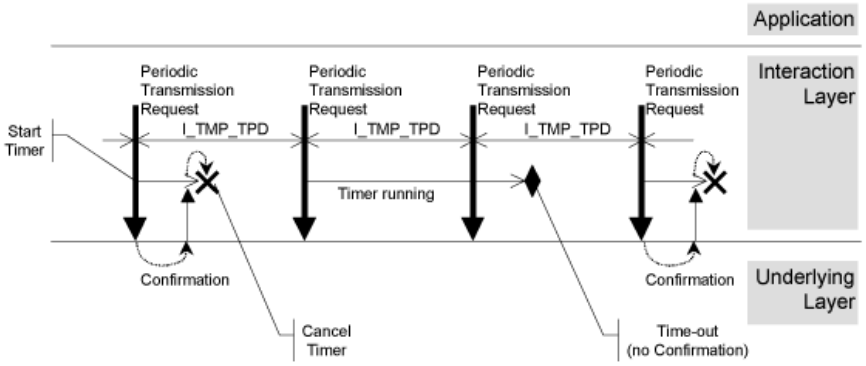

Abbildung 6: Time-Out beim Senden im periodischen Modus

Übertragungsmodus möglich. Dabei wird jeweils überwacht, ob die Nachrichten innerhalb eines festgelegten Intervalls nach der angeforderten Übertragung bzw. nach der periodischen Übertragung empfangen werden. Sollte eine Deadline nicht eingehalten werden, kann optional die Anwendung oder das OSEK indirect NM über die Nichteinhaltung der Deadline informiert werden.
Abbildung 7 zeigt ein detailliertes Modell für den Ablauf von interner Kommunikation und der Übermittlung externer Nachrichten, wie es in den vorangegangenen Abschnitten beschrieben ist.

3 Minimale Anforderungen an untere Schichten

Die Data Link Layer bzw. die Network Layer stellen die Schichten unterhalb der Interaction Layer dar. Sie haben die Aufgabe, PDUs mit fester oder dynamischer Länge von einem Steuergerät zu einem anderen zu übertragen. Um das Versenden von Nachrichten an mehrere Empfänger zu ermöglichen sollte die untere Schicht die Broadcastübermittlung unterstützen. Neben der Übertragungsfähigkeit müssen der Interaction Layer drei Dienste zur Verfügung gestellt werden.
Es wird ein Request Service benötigt, an den aus der Interaction Layer ein Übertragungswunsch mit Steuerinformationen gestellt werden kann, so dass eine schnellstmögliche Übertragung einer I-PDU erfolgen kann. Zudem ist ein Confirmation Service nötig, um den Erfolg einer Übertragung oder aber auch einen Fehler zu melden. Zu guter letzt muss die untere Schicht über einen Indi-

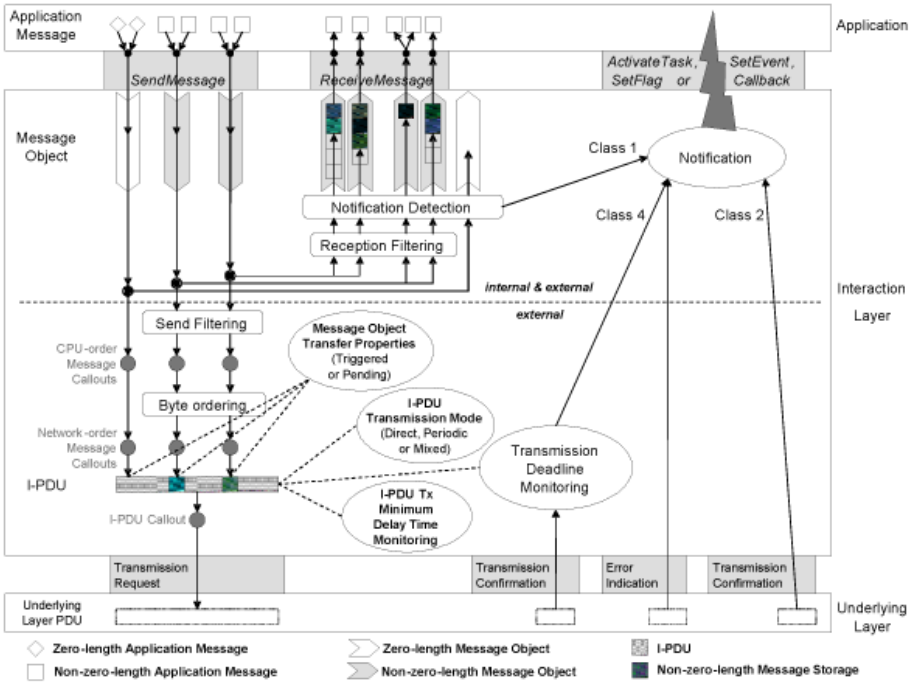

Abbildung 7: Modell für interne Kommunikation und externes Senden

cation Service, der die Interaction Layer darüber informiert, dass eine Nachricht zum Empfang vorliegt.

4 Conformance Classes

Um OSEK COM jeweils den Anforderungen an Funktionalität und den verfügbaren Hardwareressourcen anzupassen, gibt es vier verschiedene Funktionalitätsstufen in OSEK COM, die so genannten Conformance Classes. Die Klasse CCCA bietet minimale Funktionalität, sie unterstützt lediglich interne Kommunikation mit unqueued Messages. Die Klasse CCCB unterstützt ebenfalls nur die interne Kommunikation, erlaubt aber auch Queued Messages. Die minimale Funktionalität von externer Kommunikation wird von der Klasse CCC0 bereitgestellt, bei dieser Klasse sind wiederum keine Queued Messages vorgesehen. Die Klasse CCC1 schließlich unterstützt alle Features, die OSEK COM zu bieten hat. Abbildung 8 zeigt eine Übersicht über die vier Conformance Classes und die in ihnen unterstützten Features.

Features	CCCA	CCCB	CCC0	CCC1
Unqueued messages	√	√	√	√
Notification Class 1	√[5]	√	√	√
Queued messages		√		√
Message status information		√		√
External communication			√	√
Triggered Transfer Property			√	√
Notification Class 2			√	√
Byte order conversion			√	√
Direct Transmission Mode			√	√
Filtering				√
Pending Transfer Property				√
Zero-length messages				√
Dynamic-length messages				√
Periodic Transmission Mode				√
Mixed Transmission Mode				√
Minimum delay time				√
Deadline Monitoring				√
Notification Class 3				√
Notification Class 4				√
Callouts				√

Abbildung 8: Übersicht der Conformance Classes

Quellen

http://portal.osek-vdx.org/files/pdf/specs/osekcom303.pdf