AG Vernetzte Systeme
Fachbereich Informatik
Technische Universität Kaiserslautern

Masterarbeit

Drahtlose Kommunikationssysteme für den Produktionsbereich

Christopher Kramer

24. Juni 2014

Drahtlose Kommunikationssysteme für den Produktionsbereich

Masterarbeit

AG Vernetzte Systeme
Fachbereich Informatik
Technische Universität Kaiserslautern

Christopher Kramer

Tag der Ausgabe : 10.12.2013
Tag der Abgabe : 24.06.2014

Erstgutachter : Prof. Dr. Reinhard Gotzhein
Zweitgutachter : M. Sc. Dennis Christmann

Ich erkläre hiermit, die vorliegende Masterarbeit selbstständig verfasst zu haben. Die verwendeten Quellen und Hilfsmittel sind im Text kenntlich gemacht und im Literaturverzeichnis vollständig aufgeführt.

Kaiserslautern, 24. Juni 2014

(Christopher Kramer)

Zusammenfassung

Im Produktionsbereich kommen verstärkt drahtlose Kommunikationssysteme zum Einsatz, um Anlagen zu überwachen, zu steuern und zu regeln. Diese Arbeit analysiert zunächst die Anforderungen, die sich in diesem Einsatzbereich an das Kommunikationssystem ergeben, und konkretisiert sie anhand eines Anwendungsszenarios. Anschließend werden verfügbare Systeme betrachtet und insbesondere die für diesen Bereich konzipierten internationalen Standards WirelessHART und ISA 100.11a verglichen. Schließlich wird in Hinblick auf die identifizierten Anforderungen und im Vergleich zu den vorhandenen Standards ein neues drahtloses Kommunikationssystem konzipiert. Dabei wird ein dienstorientierter Ansatz verfolgt, bei dem die Anwendung über eine Middleware-Schicht auf das Kommunikationssystem zugreift. Teil dieser Middleware ist eine Service Registry zur Verwaltung der im Netzwerk verfügbaren Dienste. Die detaillierte Konzeption und Implementierung dieser Service Registry bilden den Fokus dieser Arbeit.

Abstract

In producing industries, more and more wireless communication systems are used for monitoring and controlling of production facilities. This thesis first analyzes the requirements for the communication system that yield from these applications. A concrete application scenario is presented to make the requirements more concrete. Next, available systems are examined and the international standards WirelessHART and ISA 100.11a, which have been specifically designed for these applications, are compared. Finally, a new wireless communication system is designed that meets the identified requirements and is compared to existing standards. It follows a service-oriented approach, in which applications access the communication system through a middleware layer. A service registry that manages the available services is part of this middleware. The focus of the thesis lies on the detailed conceptual design and implementation of this service registry.

Inhaltsverzeichnis

1. KAPITEL

Einleitung

Der heute herrschende starke Wettbewerbsdruck zwingt produzierende Wirtschaftsunternehmen, ihre Produktionsabläufe stetig zu optimieren, um die Effizienz zu steigern. Mit diesem Ziel werden immer mehr Produktionsabläufe automatisiert und die Automatisierung selbst immer weiter optimiert. Dabei kommen *Sensoren* zum Einsatz, die unterschiedliche Messwerte der aktuellen Produktionsabläufe, wie z.B. Temperatur oder Druck, erfassen. Im einfachsten Fall dienen diese Messwerte zur *Überwachung* der Produktion. Deuten sie auf Probleme in der Produktion hin, kann diese beispielsweise angehalten werden, um das Problem zu beseitigen. Darüber hinausgehend können die Messwerte zur *Steuerung* der Produktion dienen. Dabei liest das Überwachungspersonal die Messwerte ab und passt bei Bedarf Produktionsparameter an. So kann gewährleistet werden, dass die Produktion sicher und effizient betrieben wird. Werden die Produktionsparameter nicht mehr von Menschen, sondern vollautomatisch basierend auf den Messwerten angepasst, spricht man von *Regelung*.
Zur Überwachung und Steuerung ist es noch denkbar, dass die Messwerte von Personal abgelesen werden und darauf basierend Entscheidungen getroffen werden. Allerdings ist dies sehr aufwendig und fehleranfällig, insbesondere dann, wenn sehr viele Sensoren abzulesen sind. Deutlich einfacher, schneller und zuverlässiger ist es, die Messdaten über ein Kommunikationssystem zu übertragen. So können sie beispielsweise von einzelnen Sensoren über aggregierende Zwischenknoten zu einem zentralen Leitstand übertragen werden, wo sie angezeigt und ausgewertet werden. Das Überwachungspersonal am Leitstand kann basierend auf diesen Daten die Produktionsparameter beeinflussen, beispielsweise die Geschwindigkeit eines Motors oder einer Pumpe verändern. Diese Steuerungsbefehle würden wieder über ein Kommunikationssystem zu den *Aktuatoren*, wie etwa einem Motor oder einer Pumpe, übertragen. Im Falle der Regelung gibt das System automatisch Steuerungsbefehle an die Aktuatoren, ohne dass der Eingriff des Steuerungspersonals nötig ist.
Traditionell kommen in der Automatisierungstechnik drahtgebundene Kommunikationssysteme zum Einsatz. Mit dem Aufkommen drahtloser Kommunikationssysteme bietet sich die Möglichkeit, auch Automatisierungstechnik im industriellen Umfeld drahtlos kommunizieren zu lassen, was einerseits eine ganze Reihe Vorteile verspricht, andererseits auch viele Herausforderungen mit sich bringt [AGB11, PW10].

Die möglichen *Kosteneinsparungen* sind sicher einer der treibenden Vorteile. Drahtlose Systeme sind einfacher und günstiger zu installieren als drahtgebundene, da das aufwendige Verlegen von Kabeln entfällt. So geben Akerberg, Gidlund und Björkman an, dass das Verlegen von Kabeln in einer gewöhnlichen verfahrenstechnischen Anlage ungefähr $200 je Meter kostet

[AGB11]. In [MTA05] wird abgeschätzt, dass die Verkabelung eines Geräts mit einem drahtgebundenen Kommunikationssystem inkl. Arbeitskosten ca. $350 kostet und damit bei einer Anlage mit 200 Geräten Kosten in Höhe von $70.000 entstehen. Für ein auf ZigBee basierendes drahtloses Kommunikationssystem werden die Kosten für die Funkknoten und deren Installation mit $60 je Knoten beziffert. Dazu kommen die Anschaffungskosten eines Access Points in Höhe von ca. $100. Anschaffung und Installation eines Systems mit 200 Geräten kosten daher ca. $12.000 [MTA05]. Diese Abschätzung zeigt deutlich das Kosteneinsparungspotenzial, das hinter der Einführung drahtloser Kommunikationssysteme im Produktionsbereich steckt. Dazu kommt noch, dass davon ausgegangen wird, dass die Kosten für Wartung und Instandhaltung von Kabeln deutlich teurer sind, beispielsweise weil Kabel brechen oder Steckverbindungen sich lösen. Nach [Ene02] sind die meisten Netzwerkausfälle auf Probleme an den Steckverbindungen zurückzuführen, was durch drahtlose Systeme gelöst wird.

Dadurch, dass drahtlose Kommunikationssysteme eine einfache Überwachung und Regelung von Produktionsanlagen erlauben, ermöglichen sie auch *Energieeinsparungen* und damit zusätzliche Kosteneinsparungen. Die U.S. Energiebehörde schätzte 2006, dass indem die manuelle Überprüfung von industriell eingesetzten Motoren durch dauerhafte drahtlose Überwachung ersetzt wird, deren Energieverbrauch um bis zu 18% gesenkt werden könne [US 06].

Da drahtlose Systeme keine Kabel-Zuleitung benötigen, bieten sie eine erhöhte *Flexibilität*. So können sekundäre Prozessvariablen, die lange Zeit nicht überwacht worden sind, weil es nicht ökonomisch gewesen wäre, oder Messdaten von rotierenden Teilen, bei denen es technisch nicht möglich ist Kabel zu verlegen, mit Hilfe drahtloser Kommunikationssysteme überwacht werden [AGB11]. Indem mehr Variablen gemessen werden können, lässt sich der Produktionsprozess besser optimieren und somit die Effizienz steigern. Da drahtlose Systeme auch problemlos Messwerte von beweglichen und rotierenden Teilen erfassen können, kann durch die stetige Überwachung auch die Zuverlässigkeit und Sicherheit dieser Systeme verbessert werden. Außerdem erlauben es drahtlose Systeme, schnell und einfach temporär Messungen durchzuführen, ohne dass erst aufwendig Kabel verlegt werden müssen. Ebenso können mobile Arbeiter oder Roboter leicht drahtlos kommunizieren.

Unter Umständen ist es möglich, durch drahtlose Systeme auch die *Verfügbarkeit* und *Zuverlässigkeit* zu erhöhen, indem beispielsweise mehrere redundante Sensoren platziert werden oder die Kommunikation nicht durch Kabelbruch gestört wird. Durch den Einsatz von Mesh-Netzwerken ist es denkbar, dass bei Ausfall eines Knotens alternative Routen genutzt werden können, um gebrochene Links zu kompensieren.

Die restliche Arbeit gliedert sich wie folgt: Zunächst analysiert Kapitel 2 die Anforderungen, die im Produktionsbereich an ein drahtloses Kommunikationssystem gestellt werden. Daraufhin betrachtet Kapitel 3 vorhandene Systeme und vergleicht insbesondere die beiden wichtigsten Standards WirelessHART und ISA 100.11a. In Kapitel 4 wird anschließend ein neues drahtloses Kommunikationssystem für den Produktionsbereich konzipiert. Dabei werden die zuvor ermittelten Anforderungen beachtet und das neue System mit den existierenden Standards verglichen. Kapitel 5 geht genauer auf den Middleware Layer ein, der einen Teil des neu konzipierten Systems darstellt. Dabei wird insbesondere die im Rahmen diese Arbeit implementierte Service Registry detailliert beschrieben. Abschließend fasst Kapitel 6 die Ergebnisse dieser Arbeit zusammen und gibt einen Ausblick auf künftige Arbeiten.

2. KAPITEL

Anforderungen an drahtlose Kommunikationssysteme im Produktionsbereich

Drahtlose Kommunikationssysteme sind uns aus dem Alltag schon lange bekannt, seien es Mobiltelefone oder drahtlos über Wireless Local Area Network (WLAN IEEE 802.11) [IEE12a] ans Internet angebundene PCs. Aber obwohl es für diese Einsatzzwecke bereits bewährte standardisierte Technologien gibt, so gelten im Produktionsbereich doch deutlich andere Anforderungen als beispielsweise bei Unterhaltungselektronik. Bevor in Kapitel 3 die unterschiedlichen Standards betrachtet werden, die für die drahtlose Kommunikation im Produktionsbereich nutzbar sind, werden hier zunächst die wichtigsten Anforderungen definiert, die für derartige Systeme im Produktionsbereich gelten. Nachdem anhand eines Anwendungsszenarios der Einsatzbereich genauer dargestellt wird, werden beispielhaft für dieses Szenario Anforderungen an die Dienstgüte (Quality of Service - QoS) des Kommunikationssystems definiert und abschließend formalisiert.

Insgesamt werden in diesem Kapitel die Anforderungen aus Anwendungssicht betrachtet. Welche Anforderungen sich daraus an die einzelnen Schichten der Realisierung ableiten und wie diese umgesetzt werden, wird bei der Konzeption in Kapitel 4 behandelt.

2.1 Anwendungsszenario

Die Definition eines beispielhaften Anwendungsszenarios macht einerseits anschaulich, wie die hier betrachteten drahtlosen Kommunikationssysteme im Produktionsbereich konkret eingesetzt werden können. Zum anderen ermöglicht es uns, die Anforderungen an ein geeignetes Kommunikationssystem anhand konkreter Zahlen überprüfbar zu definieren.

Im Anwendungsszenario kommt das drahtlose Kommunikationssystem in einer Produktionsanlage zur Überwachung, Steuerung, Regelung und Notfall-Abschaltung zum Einsatz. Die Anlage hat eine Größe von ca. 300 Meter Länge und 150 Meter Breite. In der Produktionsanlage werden Waren durch Maschinen und Roboter produziert, wobei Maschinen stets stationär sind, einzelne Roboter sich dagegen in der Fabrik bewegen können. Ungefähr 40 drahtlose Funkknoten sind in der Produktionsanlage montiert, um diverse Sensordaten zu ermitteln. Einige dieser

Funkknoten sind mit mehreren Sensoren ausgestattet. Insgesamt werden Daten von 70 Sensoren erfasst. Die eingesetzten Sensoren lassen sich in neun Typen einteilen, welche in Tab. 2.1 definiert sind. Dabei gibt es zum einen periodisch getriggerte Sensoren, deren Messwert in einem festen Intervall genommen und übermittelt wird. Daneben gibt es ereignisgesteuerte Sensoren, die nicht regelmäßig, sondern nur bei Eintritt eines bestimmten sporadischen Ereignisses einen Wert übermitteln. Je nach Sensor unterscheidet sich die Datengröße eines Messwertes und wie häufig die Werte übermittelt werden müssen. Da ereignisgesteuerte Sensornachrichten kein festes Intervall haben, lässt sich hier nur angeben, wie oft das Ereignis üblicherweise auftritt. Wichtiger ist jedoch das minimale Ereignis-Eintrittsintervall Int_{min}, welches die kürzeste mögliche Zeit zwischen zwei Eintritten des Ereignisses angibt.

Tabelle 2.1.: Sensortypen im Anwendungsszenario

Typ	Trigger	Datengröße	Intervall	Int_{min}	Anzahl
Temperatur	Periodisch	32 Bit	1 s		10
Druck	Periodisch	32 Bit	1 s		6
Gaserkennung	Periodisch	8 Bit	1 s		2
Vibration	Periodisch	8 Bit	10 s		10
Drehzahl	Periodisch	16 Bit	0,1 s		10
Lichtschranke	Ereignis	1 Bit	$\sim$ 1-30 s	1 s	10
Produktfertigstellung	Ereignis	8 Bit	$\sim$ 20-50 s	20 s	1
Maschinenzustand	Periodisch	4 Bit	10 s		20
Roboterposition	Periodisch	32 Bit	10 s		1

Sämtliche Sensordaten werden zusätzlich zum Messwert mit einem Zeitstempel mit einer maximalen Abweichung von 10 ms (32 Bit) sowie einer eindeutigen Kennung (16 Bit) versehen. Die Sensordaten werden zur Überwachung und Steuerung der Anlage, teils über aggregierende Knoten, zu einem Leitstand übermittelt. Dort befindet sich ein drahtloser Knoten, der die Daten empfängt. Das Kontrollpersonal kann die Daten am Leitstand überprüfen und bei Bedarf Steuerungsbefehle an die Anlage schicken. Dazu sind in der Anlage 20 drahtlose Knoten an den Maschinen und Robotern (Aktuatoren) installiert, die diese Steuerungsbefehle empfangen und an die Maschinen bzw. Roboter weitergeben. Die möglichen Steuerungsbefehle lassen sich in die in Tab. 2.2 aufgeführten vier Typen einteilen.

Tabelle 2.2.: Typen von Steuerungsbefehlen im Anwendungsszenario

Typ	Datengröße	Intervall	Int_{min}	Anzahl
Ventilöffnung / Schließung	8 Bit	$\sim$ 1-3 h	30 s	4
Maschinenabschaltung	1 Bit	Selten	60 s	10
Zielposition Roboter	32 Bit	$\sim$ 30-90 min	30 s	1
Heizungs-Solltemperatur	16 Bit	$\sim$ 20-120 min	30 s	5

Steuerungsbefehle werden genau wie Sensordaten mit einer eindeutigen Kennung (16 Bit), allerdings nicht mit einem Zeitstempel, versehen. Alle Steuerungsbefehle sind ereignisgesteuert (durch das Kontrollpersonal ausgelöst). Zusätzlich zu den vom Leitstand gegebenen Steuerungsbefehlen gibt es in der Anlage auch vollautomatische Regelungssysteme, die basierend auf gemessenen Sensordaten Aktuatoren direkt ansprechen, ohne dass das Kontrollpersonal ein-

greifen muss. Es werden auf diese Weise 15 Aktuatoren automatisch geregelt, wobei vier unterschiedliche Typen von Regelungsbefehlen Verwendung finden, die in Tab. 2.3 aufgelistet sind. Das Versenden von Regelungsbefehlen kann wie bei Sensorwerten entweder periodisch in einem festen Intervall oder ereignisgesteuert, z.B. bei Überschreitung eines Grenzwerts, ausgelöst werden. Die meisten Aktuatoren führen Regelungsalgorithmen nicht selbst aus, sondern bekommen die Regelungsbefehle von einem Reglerknoten, der Sensorwerte von einem oder mehreren Sensoren empfängt und darauf basierend Regelungswerte berechnet und Regelungsbefehle an den Aktuatorknoten sendet. Dazu kommen im Netzwerk 10 weitere Funkknoten als Regler zum Einsatz.

Tabelle 2.3.: Typen von Regelungsbefehlen im Anwendungsszenario

Typ	Trigger	Datengröße	Intervall	Int_{min}	Anzahl
Motorgeschwindigkeit	Periodisch	16 Bit	1 s		5
Ventilöffnung / Schließung	Ereignis	8 Bit	$\sim$ 1 min	10 s	4
Maschinen-Notabschaltung	Ereignis	1 Bit	Selten	60 s	4
Heizungs-Solltemperatur	Periodisch	16 Bit	5 min		2

2.2 Performanz

Die Performanz eines drahtlosen Kommunikationssystems setzt sich aus zwei Aspekten zusammen: der Übertragungsrate, also der Geschwindigkeit, mit der Daten übermittelt werden können, sowie der Übertragungsverzögerung, also der Zeit, die die Übertragung eines Bits von einem Knoten zum anderen benötigt.

Übertragungsrate Die Übertragungsrate eines Knotens für Sensornetzwerke kann, verglichen mit anderen Kommunikationssystemen, beispielsweise aus der Unterhaltungselektronik, verhältnismäßig gering ausfallen. Da die Datenmengen der Nachrichten im Vergleich gering sind (s. Anwendungsszenario in Kapitel 2.1), wird keine hohe Übertragungsrate benötigt. Typische übertragene Daten sind Sensorwerte und Steuerungsbefehle im Umfang von wenigen Bytes bei Updatefrequenzen im Bereich ab 100 ms. Im ungünstigsten Fall können nie zwei Knoten gleichzeitig senden, weil jeder Knoten jeden anderen beim Senden stört. Im Anwendungsszenario benötigt dann ein drahtloses Kommunikationssystem mindestens eine Brutto-Übertragungsrate von ca. 185 kbit/s, um die geforderten Anforderungen einhalten zu können (s. Anhang A). Um noch etwas Spielraum nach oben zu lassen, wird eine minimale Brutto-Übertragungsrate von 200 kbit/s verlangt.

Übertragungsverzögerung Wichtiger als die Übertragungsrate ist für ein Kommunikationssystem im industriellen Umfeld die Übertragungsverzögerung. Aus Sicherheitsgründen muss beispielsweise bei einem Regelungsbefehl zur Maschinen-Notabschaltung in jedem Fall eine maximale Verzögerung eingehalten werden; eine unbegrenzte maximale Verzögerung, wie sie z.B. in der Unterhaltungselektronik vorkommen darf und zum Verwerfen der Daten mit der Folge von Qualitätseinbußen führt, wäre hier undenkbar.

Damit der Schedule periodischer Nachrichten eingehalten werden kann und sich die Nachrichten nicht aufstauen, soll die maximale Verzögerung stets geringer als das Update-Intervall sein.

Darüber hinaus darf bei periodischen Nachrichten eine maximale Verzögerung von einer Sekunde nicht überschritten werden. Ausnahmen sind die Nachrichten des Gaserkennungs-Sensors sowie die Regelungsbefehle zur Regelung der Motorgeschwindigkeit, welche maximal 500 ms verzögert werden dürfen.
Bei den ereignisgesteuerten Nachrichten hängt die maximale Verzögerung stark vom Typ der Nachricht ab. Die zulässigen maximalen Verzögerungen sind in Abhängigkeit des Nachrichtentyps in Tab. 2.4 aufgeführt.

Tabelle 2.4.: Maximale Verzögerungen für ereignisgesteuerte Nachrichten

Nachrichtentyp	maximale Verzögerung (s)
Sensor Produktfertigstellung	5,0
Sensor Lichtschranke	1,0
Steuerungsbefehle	1,0
Regelung Ventil	0,5
Regelung Notabschaltung	0,1

2.3 Zuverlässigkeit

Zuverlässigkeit ist im industriellen Einsatz eine besonders wichtige Anforderung. Da drahtlose Kommunikationssysteme aber auf einem potenziell unzuverlässigen Medium aufbauen, können Verluste und Verfälschungen durch das Medium nicht ausgeschlossen werden. Gerade im Produktionsbereich muss sowohl damit gerechnet werden, dass andere drahtlose Kommunikationssysteme – wie etwa WLAN oder Bluetooth [Blu13b] – das Medium mitbenutzen, als auch dass Störungen durch Maschinen, wie etwa Motoren, Thermostate oder Computer auftreten [LWE05]. Um trotzdem die gerade im industriellen Einsatz benötigte Zuverlässigkeit zu erreichen, muss das Kommunikationssystem Verluste und Verfälschungen erkennen und eventuell entsprechende Maßnahmen ergreifen, um diese auszugleichen.

Verluste Obwohl Verluste auf einem drahtlosen Medium nicht ausgeschlossen werden können, gibt es dennoch Mechanismen, um die Wahrscheinlichkeit von Verlusten zu reduzieren, beispielsweise indem eine freie Frequenz genutzt und so Interferenzen mit anderen Systemen vermieden werden. Daher ist es zunächst wünschenswert, dass das Kommunikationssystem geeignete Maßnahmen einsetzt, um die Verlustrate bestmöglich zu reduzieren. Kommt es dennoch zu Verlusten, so muss das Kommunikationssystem dies in jedem Fall erkennen. Wie das System auf den Verlust reagiert, hängt vom zugehörigen Nachrichtentyp ab.

- Steuerungs- und Regelungsbefehle sowie ereignisgesteuerte Sensor-Nachrichten müssen neu übertragen werden, bis die Übertragung erfolgreich war oder nach 20 Sekunden ein Timeout auftritt. Wird ein Timeout ausgelöst, so muss das System automatisch in einen sicheren Zustand gefahren werden.
- Periodische Sensor-Nachrichten werden bei Verlust verworfen, es findet also keine Neuübertragung statt. Allerdings dürfen die Nachrichten nicht beliebig oft in Folge verloren gehen, sonst muss das System automatisch in einen sicheren Zustand fahren.
  - Nachrichten der Sensoren Gaserkennung und Maschinenzustand dürfen maximal 5x in Folge verloren gehen.

– Nachrichten der anderen Sensoren dürfen maximal 20x in Folge verloren gehen.

Verfälschungen Die Verfälschung einer Nachricht muss erkannt werden. Sie führt dazu, dass die Nachricht verworfen wird. Die Behandlung erfolgt dann so, als sei ein Verlust aufgetreten.

Umordnung Er muss sichergestellt sein, dass die Verarbeitung von Steuerungs- und Regelungsbefehlen exakt in der Reihenfolge erfolgt, wie sie versendet wurden. Bei Sensornachrichten ist eine Umordnung zulässig, da die Werte ohnehin anhand des Zeitstempels geordnet werden können.

2.4 Garantien

In der industriellen Anwendung ist es besonders wichtig, dass die Anforderungen, die an Performanz und Zuverlässigkeit des Systems gestellt werden, mit definierten Garantien eingehalten werden. Da es oft um die Überwachung und Steuerung sicherheitsrelevanter Systeme geht, ist eine *Best-Effort* Dienstgüte [CAH96], die keinerlei Garantien für die Einhaltung von Anforderungen gibt, völlig unzureichend.
Für ein drahtloses Kommunikationssystem ist es allerdings extrem schwierig, eine *deterministische* Dienstgüte, also die garantierte Einhaltung der geforderten Anforderungen, gewährleisten zu können. Dies liegt daran, dass das zugrundeliegende drahtlose Medium auch von anderen Systemen genutzt oder beispielsweise durch Maschinen gestört werden kann, und somit prinzipiell unzuverlässig ist. Wird das drahtlose Medium auf allen zur Verfügung stehenden Frequenzen gestört, so bleibt dem drahtlosen Kommunikationssystem keine Möglichkeit mehr, Daten erfolgreich zu übertragen, um Garantien einzuhalten. Handelt es sich um ein sicherheitskritisches System, so bleibt in diesem Fall keine andere Option, als es kontrolliert in einen sicheren Zustand zu bringen. Dazu ist es erforderlich, dass das drahtlose Kommunikationssystem Störsituationen erkennt und der Befehl, das System in einen sicheren Zustand zu fahren, trotz gestörtem Funkmedium alle Knoten zuverlässig erreicht bzw. aufgrund lokaler Informationen ausgelöst wird.
Unter der Voraussetzung, dass keine Interferenzen durch externe Störquellen auftreten, soll das Kommunikationssystem sämtliche Anforderungen an Performanz und Zuverlässigkeit – d.h. minimale Übertragungsrate, maximale Übertragungsverzögerung, Behandlung von Verlusten, Verfälschungen sowie Umordnung – *deterministisch* erfüllen. Treten Interferenzen durch externe Störquellen auf, die dazu führen, dass Anforderungen nicht eingehalten werden können, muss das System diese erkennen und das System kontrolliert in einen sicheren Zustand bringen.

2.5 Skalierbarkeit

Die Skalierbarkeit des Kommunikationssystems kann im Produktionsbereich ebenfalls eine wichtige Rolle spielen. Wird beispielsweise eine Produktionsanlage ausgebaut, so werden auch dem drahtlosen Kommunikationssystem neue Funkknoten hinzugefügt. Das Kommunikationssystem sollte also in einem gewissen Maße in Bezug auf die Netzwerkgröße skalierbar sein. Allerdings wird hier nicht davon ausgegangen, dass zu einem drahtlosen Netzwerk einer Produktionsanlage 1000 oder mehr Knoten gehören. Obwohl derart hohe Anforderungen an die Skalierbarkeit solcher Systeme in der Forschung genannt werden [GH09], wird von ABB [ABB], einem

Konzern, der unter anderem auf die Automatisierung von Industrieanlagen spezialisiert ist, angegeben, dass ca. 50 Knoten in einem drahtlosen Netzwerk einer Produktionsanlage realistisch sind [AGB11]. Es ist auch nicht zwingend erforderlich, dass sämtliche Funkknoten einer Produktionsanlage im selben Funknetzwerk verbunden sind. Unterschiedliche Produktionsstraßen können unterschiedliche Funknetzwerke betreiben und bei Bedarf über Gateways Daten untereinander austauschen. Eine derart hierarchische Netzwerkstruktur ist nicht nur äußerst skalierbar, sie bietet auch Vorteile in Organisation und Wartung.

In unserem Anwendungsszenario gehen wir von 40 Sensorknoten, 20 Aktuatorknoten und 10 Reglerknoten, also insgesamt 70 Knoten in einem Funknetzwerk aus. Um noch etwas Raum für Erweiterungen zu lassen, verlangen wir vom Kommunikationssystem, mindestens 100 Knoten in einem drahtlosen Netzwerk zu unterstützen. Darüber hinaus muss es möglich sein, über Gatewayknoten mit anderen drahtlosen oder drahtgebundenen Netzwerken zu kommunizieren.

Da die Reichweite von typischen Funkknoten im Innenbereich je nach Technologie meist nicht mehr als 10 m bis 30 m beträgt (s. Kapitel 3) und die Größe einer Produktionsanlage wie im Anwendungsszenario leicht 100 m übersteigt, muss das Kommunikationssystem *Multi-Hop-Topologien* unterstützen. Im Gegensatz zu einer *Single-Hop-Topologie*, bei der wie in Abb. 2.1 illustriert jeder Knoten mit jedem anderen in direkter Verbindung steht, müssen Nachrichten bei einer Multi-Hop-Topologie unter Umständen von Zwischenknoten, sogenannten *Routern*, weitergeleitet werden.

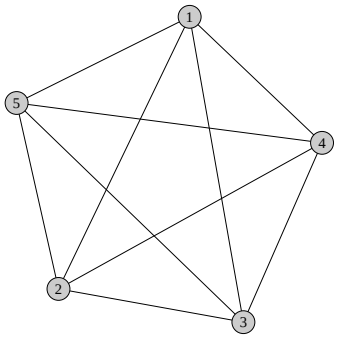


Abbildung 2.1.: Single-Hop-Topologie mit 5 gleichwertigen Knoten

Die *Sterntopologie* ist eine der einfachsten Multi-Hop-Topologien: Hierbei haben alle Endgeräte eine direkte Verbindung zu einem zentralen Router und jedes Paket wird zuerst zum Router gesendet und von dort zum Zielknoten weitergeleitet (s. Abb. 2.2). Da die Sterntopologie die maximal mögliche Entfernung der Endgeräte lediglich verdoppelt, können so mit typischen Funkknoten maximal 20 bis 60 m im Innenbereich erzielt werden. Da dies nicht ausreicht, um Produktionsanlagen mit Entfernungen von 100 m und mehr abzudecken, ist diese Topologie für das betrachtete Szenario unzureichend. Insgesamt skaliert die Sterntopologie nicht, da alle Knoten mit nur einem zentralen Router verbunden sind.

Bei der *Mesh-Topologie* übernimmt dagegen jeder Knoten die Funktion eines Routers und kann Pakete für andere Knoten weiterleiten (s. Abb. 2.3). Die Knoten können dabei beliebig verbunden sein, wodurch es möglich wird, beliebig große Topologien zu bilden. Selbst sehr weitläufige Produktionsanlagen können so durch Platzierung von genügend Routern abgedeckt werden.

Ein Kompromiss aus Stern- und Mesh-Topologie stellt die *Stern-Mesh-Topologie* dar (s. Abb. 2.4), bei der die Router eine Mesh-Topologie bilden und die Endgeräte sternförmig mit den Routern verbunden sind. Auch bei dieser Topologie können beliebig große Netzwerke gebildet werden, sofern genügend Router vorhanden sind.

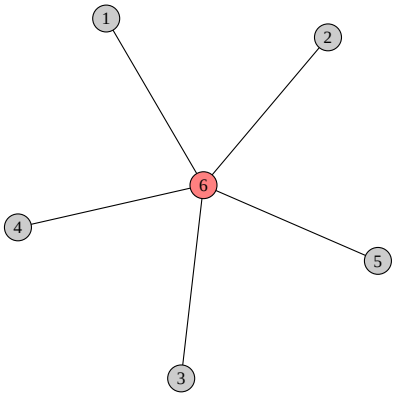


Abbildung 2.2.: Sterntopologie mit 5 Endgeräten (grau) und einem zentralen Router (rot)

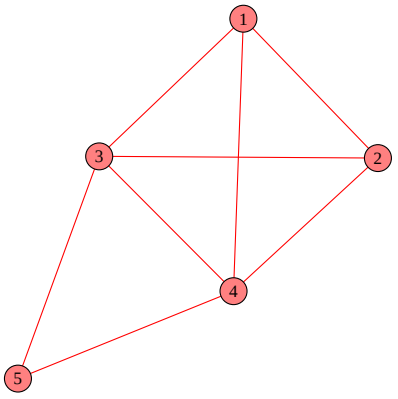


Abbildung 2.3.: Mesh-Topologie aus 5 Routern

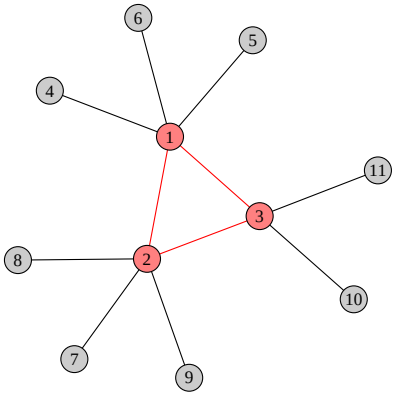


Abbildung 2.4.: Stern-Mesh-Topologie mit 3 als Mesh vernetzten Routern (rot) und 8 sternförmig angebundenen Endgeräten (grau)

Aus Anwendungssicht ist es nicht relevant, welche Multi-Hop-Topologie das Kommunikationssystem bildet. Es ist nur wichtig, dass die benötigte Entfernung abgedeckt werden kann. Daher sind Mesh- oder Stern-Mesh-Topologien denkbar. Welche Vor- und Nachteile diese in Bezug auf die Einhaltung der anderen Anforderungen haben, wird in Kapitel 4 genauer untersucht werden.

2.6 Flexibilität

Eine weitere wichtige Anforderung ist die Flexibilität des Systems. Wie im Anwendungsszenario beschrieben, kann es in einer Produktionsanlage Roboter geben, die sich innerhalb der Anlage bewegen können. Durch die Positionsänderung kann sich die Topologie des Funknetzwerks ändern. Das drahtlose Kommunikationssystem kann also nicht von einer völlig statischen Topologie ausgehen. Allerdings ist es auch nicht nötig, dass eine völlig dynamische Topologie, bei der sich alle Knoten jederzeit bewegen können, unterstützt wird. In einer Produktionsanlage sind die meisten Sensoren und Aktuatoren typischerweise fest installiert und nur wenige Knoten mobil. Um sowohl die mehrheitlich stationären als auch die wenigen mobilen Knoten optimal anzubinden, sollte es das Kommunikationssystem idealerweise erlauben, zu konfigurieren, ob ein Knoten stationär oder mobil ist.

Ein weiterer Aspekt der Flexibilität stellt die Erweiterbarkeit des Systems dar. Obwohl im Anwendungsszenario eine sehr begrenzte Menge möglicher Knotentypen, Steuerungs- und Regelungsbefehle angegeben ist, so ist es leicht denkbar, dass neue Typen hinzukommen. Beispielsweise könnte die Produktionsanlage mit zusätzlichen Helligkeitssensoren ausgerüstet werden. Die Anbindung dieser neuartigen Knoten an das Netzwerk sollte möglich sein, ohne dass die Anlage dazu heruntergefahren und das Netz neu konfiguriert werden muss. Es wird daher ein flexibler Mechanismus verlangt, mit dem *Dienste*, die von einem Knoten bereitgestellt werden, während des Betriebs bekannt gemacht werden können, sodass andere Knoten sie auffinden und abonnieren können. Ein Dienst könnte das Bereitstellen eines Sensorwertes oder das Entgegennehmen von Befehlen zur Steuerung bzw. Regelung eines Aktuators sein.

Die Zuordnung von Diensten zu Knoten sollte ebenfalls flexibel gestaltbar sein. Da, wie im Anwendungsszenario erwähnt, ein Funkknoten mit mehreren Sensoren ausgestattet sein kann, muss das System es erlauben, dass ein Knoten mehrere Dienste anbietet. Ebenso muss es möglich sein, dass ein Knoten mehrere Dienste abonniert. So könnte beispielsweise ein Reglerknoten zur Regelung einer Heizung die Sensorwerte mehrerer Temperatur-Sensoren abonnieren, um anhand dieser Daten die Soll-Temperatur einer Heizung zu berechnen und als Regelungsbefehl an den Aktuator-Knoten der Heizung zu senden. Gleichermaßen könnte ein Knoten als *virtueller Sensorknoten* die Sensorwerte mehrerer Sensoren abonnieren und sie aggregiert als Dienst bereitstellen, der von mehreren anderen Knoten abonniert werden kann.

Eine weitere Flexibilitäts-Anforderung ergibt sich aus der Wartbarkeit des Systems. Da das Anhalten einer Industrieanlage Kosten verursacht, sollte das drahtlose Kommunikationssystem so flexibel sein, dass es häufige Wartungsarbeiten während des Betriebs zulässt. Fällt beispielsweise ein unkritischer Sensorknoten aus, so sollte es möglich sein, ihn durch einen neuen zu ersetzen, ohne die Anlage anhalten zu müssen. Ebenso sollte es während des Betriebs möglich sein, zusätzliche Knoten zum Netzwerk hinzuzufügen.

2.7 Sicherheit

In drahtlosen Kommunikationssystemen ist es gegenüber drahtgebundenen Systemen schwieriger, die Sicherheit des Systems zu gewährleisten. Da auf das drahtlose Medium auch von außerhalb der Anlage zugegriffen werden kann, können Angreifer die gesendeten Nachrichten potenziell abhören, gefälschte Nachrichten senden oder durch Störsignale die Verfügbarkeit des Systems beeinträchtigen. Ein solcher Angriff auf das Kommunikationssystem kann Kosten verursachen oder die Betriebssicherheit der Anlage gefährden. Daher müssen geeignete Maßnahmen ergriffen werden, um Angriffe abzuwehren. Dabei müssen mehrere Aspekte der Sicherheit berücksichtigt werden:

- *Vertraulichkeit*: Es muss sichergestellt sein, dass Nachrichten von einen Angreifer nicht abgehört werden können.
- *Integrität*: Es muss sichergestellt sein, dass ein Angreifer Nachrichten nicht modifizieren kann bzw. dass eine Modifikation erkannt wird.
- *Authentizität*: Es muss sichergestellt sein, dass zu jeder empfangenen Nachricht überprüft werden kann, ob sie vom angegebenen Absender stammt oder nicht.
- *Verfügbarkeit*: Da nicht ausgeschlossen werden kann, dass ein Angreifer das drahtlose Medium durch einen Störsender blockiert und das System damit unverfügbar macht, müssen Maßnahmen getroffen werden, dies zu erkennen und das System in einen sicheren Zustand zu bringen.

Wie das drahtlose Kommunikationssystem diese Aspekte gewährleistet, ist aus Anwendungssicht nicht relevant. Wichtig ist nur, dass nach aktueller Ansicht sichere Verfahren zum Einsatz kommen, also beispielsweise nicht auf gebrochene Verschlüsselungsverfahren vertraut wird.

2.8 Ressourcen-Effizienz

Einer der Vorteile drahtloser Kommunikationssysteme ist es, dass Knoten sich bewegen können, da sie nicht an ein festes Kabel gebunden sind. Damit dieser Vorteil genutzt werden kann, ist es erforderlich, dass die Knoten ohne feste Energieversorgung auskommen, also mit einer Batterie oder einem Akku betrieben werden können. Damit Batterien nicht zu häufig ausgetauscht werden müssen, sollte die Lebensdauer einer Batterie mindestens einige Monate, besser Jahre betragen. Da in einer Produktionsanlage allerdings auch viele stationäre Knoten problemlos an eine feste Energieversorgung angeschlossen werden können, sollte das Kommunikationssystem idealerweise eine Unterscheidung zwischen Knoten mit fester Energieversorgung und Knoten mit Batterieversorgung erlauben. So ist es möglich, dass Knoten mit fester Energieversorgung mehr Aufgaben übernehmen und damit die Knoten entlasten, die von einer Batterie gespeist werden.

Sollen Knoten beweglich sein, so ergeben sich damit auch Anforderungen an Größe und Gewicht der Knoten. Es ist in einer Produktionsanlage zwar i.d.R. nicht nötig, dass die Knoten Briefmarkengröße haben, allerdings sollte es möglich sein, Knoten zu fertigen, die bei einer Dicke von bis zu 5 cm nicht deutlich größer als eine Postkarte sind. So ist sichergestellt, dass die Knoten problemlos in der Nähe von Sensoren und Aktuatoren in der Anlage platziert werden können.

Da die Funkknoten mit wenig Energie auskommen, klein und möglichst günstig sein sollen, ist die eingesetzte Hardware verhältnismäßig schwach. Daher ist das Kommunikationssystem

gezwungen, effizient mit der zur Verfügung stehenden Rechenleistung sowie dem zur Verfügung stehenden persistenten und flüchtigen Speicher umgehen.

2.9 QoS-Anforderungen Applikation

Nachdem die Anforderungen an das Kommunikationssystem in Bezug auf unterschiedliche Aspekte beschrieben wurden, sollen nun die QoS-Anforderungen von drei exemplarischen Datenströmen formal definiert werden. Dabei kommt der Ansatz zur formalen Spezifikation von QoS-Anforderungen für Netzwerke aus [WG07] zum Einsatz.

Zu jedem Datenstrom wird eine QoS-Spezifikation $qosReq$ angegeben, welche sich aus q_{base}, q_{pref} und s zusammensetzt:
$qosReq = (q_{base}, q_{pref}, s)$

- q_{base} spezifiziert die minimale Netzwerkgüte, die für eine ausreichende Anwendungsgüte erforderlich ist.
- q_{pref} spezifiziert die Netzwerkgüte, die für eine bevorzugte Anwendungsgüte gewünscht ist.
- s beschreibt die Skalierung anhand von Nutzen, Kosten und Skalierungs-Schwellwerten, wird im Folgenden allerdings nicht genauer spezifiziert.

Um q_{base} und q_{pref} angeben zu können, werden zunächst Performanz-, Zuverlässigkeits- und Granatiedomäne definiert:

- Die *Performanzdomäne P* beschreibt die Effizienzaspekte des Datenstroms hinsichtlich Ressourcen und Zeit. Hier wird sie definiert als die Menge aller geordneten Tripel bestehend aus der minimalen Periodendauer $Period_{min}$, der maximalen Übertragungsverzögerung $Delay_{max}$ und der Länge einer Nachricht $Length$:

$$P = Period_{min} \times Delay_{max} \times Length$$
$$Period_{min} = \mathbb{R}^{+}[ms], Delay_{max} = \mathbb{R}^{+}[ms], Length = \mathbb{N}[b]$$

- Die *Zuverlässigkeitsdomäne R* beschreibt die Betriebssicherheit des Datenstroms. Sie wird definiert als kartesisches Produkt aus Verlust $Loss$, Periode $Period$, auf die sich die Verlustangabe bezieht, Anzahl maximal in Folge auftretender Verluste $Burstiness$ sowie der Uhrenungenauigkeit $ClockOffset$:

$$R = Loss \times Period \times Burstiness \times ClockOffset$$
$$Loss = \mathbb{N}_0, Period = \mathbb{R}^{+}[ms], Burstiness = \mathbb{N}_0, ClockOffset = \mathbb{R}^{+}[ms]$$

- Die *Garantiedomäne G* beschreibt den Grad der Verbindlichkeit. Sie wird definiert als kartesisches Produkt von Verbindlichkeitsverpflichtung DoC, dem zugehörigen Wert im Falle einer statistischen Garantie $Stat$, sowie einer Priorität $Prio$, mit der mehrere QoS-Anforderungen unterschiedlich priorisiert werden können:

$$G = DoC \times Stat \times Prio$$
$$DoC = \{bestEffort, enhancedBestEffort, statistical, deterministic\},$$
$$Stat = \{p \in \mathbb{R} | 0 < p \leq 1\}, Prio = \mathbb{N}_0$$

Nachdem diese Domänen definiert sind, werden für q_{base} und q_{pref} jeweils für alle drei Domänen Variablen definiert:

$$q_{base} = (p_{base}, r_{base}, g_{base}) \text{ mit } p_{base} \in P, r_{base} \in R, g_{base} \in G$$
$$q_{pref} = (p_{pref}, r_{pref}, g_{pref}) \text{ mit } p_{pref} \in P, r_{pref} \in R, g_{pref} \in G$$

Diese Variablen werden nun für drei beispielhafte Datenströme mit konkreten Werten, welche die Anforderungen des jeweilgen Datenstroms widerspiegeln, festgesetzt.

2.9.1 Datenstrom 1: Sensor Temperatur

Eine Nachricht dieses Datenstroms ist wie in Kapitel 2.1 beschrieben 80 Bit groß (4 Byte Temperaturwert, 4 Byte Zeitstempel, 2 Byte Kennung). Nachrichten werden periodisch jede Sekunde oder falls möglich alle 500 ms gesendet. Die maximale Übertragungsverzögerung darf wie in Kapitel 2.2 gefordert das Übertragungsintervall nicht überschreiten und nicht mehr als 1 s betragen. Für den Verlust wird wie in Kapitel 2.3 keine maximale Verlusthäufigkeit pro Zeit gefordert, allerdings eine maximale Anzahl von 20 verlorenen Nachrichten in Folge. Außerdem wird ein maximaler ClockOffset von 10 ms verlangt. Die Garantieverbindlichkeit wird mit *enhancedBestEffort* spezifiziert, was wie folgt zu verstehen ist: Unter der Annahme, dass das Medium nicht von externen Störquellen beeinträchtigt wird, müssen die Anforderungen deterministisch eingehalten werden. Kommt es zu einer Störung durch externe Quellen, ist eine deterministische Einhaltung unmöglich, daher gilt dann lediglich eine BestEffort-Verbindlichkeit. Wie bereits erwähnt, muss das System in diesem Fall die Anlage allerdings kontrolliert in einen sicheren Zustand bringen.

$$p_{base,1} = (1\ s, 1\ s, 80\ b)$$
$$p_{pref,1} = (500\ ms, 500\ ms, 80\ b)$$
$$r_{base,1} = (undefined, undefined, 20, 10\ ms)$$
$$r_{pref,1} = r_{base,1}$$
$$g_{base,1} = (enhancedBestEffort, 1, 0)$$
$$g_{pref,1} = g_{base,1}$$

2.9.2 Datenstrom 2: Sensor Produktfertigstellung

Eine Nachricht dieses Datenstroms ist wie in Kapitel 2.1 beschrieben 160 Bit groß (2 Byte Art des Produkts, 4 Byte Zeitstempel, 2 Byte Kennung). Nachrichten werden ereignisgesteuert gesendet (übliche Intervalle: alle 20-50 s), das minimale Ereignis-Eintrittsintervall Int_{min} beträgt 20 s. Wie in Tab. 2.4 angegeben, beträgt die maximal zulässige Übertragungsverzögerung für Nachrichten dieses Datenstroms 5 s. Da es sich um einen ereignisgesteuerten Datenstrom handelt, dürfen – wie in Kapitel 2.3 beschrieben – keine Verluste auftreten. Der maximale ClockOffset muss wie beim ersten Datenstrom auf 10 ms begrenzt sein. Außerdem wird genau wie beim ersten Datenstrom eine enhancedBestEffort Verbindlichkeit gefordert, die ein deterministisches Verhalten verlangt, solange keine Störung durch externe Quellen vorliegt.

$$
\begin{aligned}
p_{base,2} &= (20.000\ ms, 5000\ ms, 160\ b) \\
p_{pref,2} &= p_{base,2} \\
r_{base,2} &= (0, undefined, undefined, 10\ ms) \\
r_{pref,2} &= r_{base,2} \\
g_{base,2} &= (enhancedBestEffort, 1, 0) \\
g_{pref,2} &= g_{base,2}
\end{aligned}
$$

2.9.3 Datenstrom 3: Steuerungsbefehl Maschinenabschaltung

Eine Nachricht dieses Datenstroms ist wie in Kapitel 2.1 beschrieben 17 Bit groß (1 Bit Abschaltung, 16 Bit Kennung). Nachrichten werden ereignisgesteuert im Fehlerfall gesendet, also äußerst selten. Das minimale Ereignis-Eintritts-Intervall Int_{min} beträgt 60 s. Wie alle Steuerungs- und Regelungsnachrichten darf aus Anwendungssicht keine Nachricht verloren gehen. Auch hier wird wieder ein ClockOffset von 10 ms sowie eine enhancedBestEffort Verbindlichkeit gefordert.

$$
\begin{aligned}
p_{base,3} &= (60.000\ ms, 1.000\ ms, 17\ b) \\
p_{pref,3} &= P_{base,3} \\
r_{base,4} &= (0, undefined, undefined, 10\ ms) \\
r_{pref,3} &= r_{base,3} \\
g_{base,3} &= (enhancedBestEffort, 1, 0) \\
g_{pref,3} &= g_{base,3}
\end{aligned}
$$

3. KAPITEL

State of the Practice

Zunächst wird in diesem Kapitel ein Überblick über Standards gegeben, die in drahtlosen Kommunikationssystemen im Produktionsbereich eingesetzt werden können. Dabei werden zuerst Standards betrachtet, die nicht speziell für industrielle Produktionsanlagen, sondern ursprünglich für andere Einsatzzwecke gedacht sind. Zu jedem dieser Standards wird insbesondere untersucht, welche Stärken und Schwächen er in Hinblick auf den Einsatz im industriellen Produktionsbereich aufweist. Im Anschluss werden drei Standards vorgestellt, die speziell für den Einsatz in industriellen Produktionsanlagen entworfen worden sind.
Im zweiten Teil dieses Kapitels werden die beiden wichtigsten Standards für drahtlose Kommunikationssysteme im Produktionsbereich, WirelessHART [Eur10] und ISA 100.11a [ISA11], detailliert verglichen. Dazu werden die Gemeinsamkeiten und Unterschiede der Systeme auf den einzelnen Protokollebenen untersucht.

3.1 Überblick über existierende Standards

Es gibt einige standardisierte Protokolle für drahtlose Kommunikationssysteme, die für den Einsatz im Produktionsbereich denkbar sind. Neben Standards, die in erster Linie für vielseitige Einsatzzwecke entworfen worden sind, wie Bluetooth, WLAN, IEEE 802.15.4, ZigBee und UWB, wurden mittlerweile mit WirelessHART, ISA 100.11a und WIA-PA gleich mehrere Protokolle speziell für den Einsatz in industriellen Produktionsanlagen standardisiert. In diesem Kapitel wird ein Überblick über die genannten Standards gegeben und aufgezeigt, warum jene, die nicht speziell für den Einsatz im industriellen Produktionsumfeld konzipiert wurden, den Anforderungen nur unzureichend gerecht werden.

3.1.1 IEEE 802.15.1 Bluetooth

Bluetooth wurde bereits ab 1994 vom Mobilfunkspezialisten Ericsson mit dem Ziel entwickelt, eine günstige und energiesparende Schnittstelle zwischen Handys und Handyzubehör zu erhalten [Haa98]. Um die Verbreitung der Schnittstelle zu fördern und sie somit erfolgreicher zu machen, gründete Ericsson 1998 zusammen mit Nokia, IBM, Toshiba und Intel die Bluetooth Special Interest Group (SIG), der mittlerweile über 20.000 Unternehmen angehören [Blu14]. Basierend auf Bluetooth Version 1.1 wurde der internationale Standard IEEE 802.15.1 [IEE05b] geschaffen, der im Gegensatz zu Bluetooth nur Physical- und MAC Layer spezifiziert. Die folgenden von

der Bluetooth SIG veröffentlichten Bluetooth-Versionen werden zwar von zahlreichen Geräten unterstützt, führten allerdings bisher nicht zu einer Aktualisierung des internationalen IEEE-Standards.
Mittlerweile findet Bluetooth Verwendung vor allem in Handys, Handy-Zubehör, Laptops und Computer-Peripherie-Geräten.
Die ursprüngliche Version 1.0 erreicht eine Brutto-Datenrate von 1,0 Mbit/s. Mit Bluetooth 2.0 + EDR (Enhanced Data Rate) [Blu04] und Bluetooth 3.0 + HS (High Speed) [Blu09] wurde die maximal mögliche Brutto-Datenrate auf bis zu 24 Mbit/s erhöht (s. Tab. 3.1). Ab Version 3.0 kommt dabei ein zusätzlicher auf IEEE 802.11 (WLAN) basierender High-Speed-Kanal zum Einsatz. Version 4.0 [Blu10] unterstützt neben dem klassischen Bluetooth auch das energiesparende Bluetooth Low Energy, was allerdings nur Datenraten bis zu 1 Mbit/s erlaubt. Für die Anwendung als drahtloses Kommunikationssystem im Produktionsbereich, wie sie in Kapitel 2 spezifiziert wurde, bieten alle Bluetooth-Versionen eine mehr als ausreichende Übertragungsrate.

Tabelle 3.1.: Bluetooth Versionen und Datenraten

Version	Brutto-Datenrate (Mbit/s)	Veröffentlicht
1.0	1	1999
2.0 + EDR	3	2004
3.0 + HS	24	2009
4.0 Low Energy	1	2010

Bluetooth verwendet das lizenzfreie 2,4 GHz ISM-Band (Industrial, Scientific, Medical). Das Frequenzspreizverfahren *Frequency Hopping Spread Spectrum* kommt zum Einsatz, um die Robustheit von Bluetooth gegenüber Störungen im vielfältig genutzten 2,4 GHz Band zu verbessern. Das Verfahren wechselt bis zu 1600 mal pro Sekunde zwischen 79 verschiedenen Frequenzen. Auf diese Weise wird Bluetooth kaum gestört, wenn auf einzelnen Frequenzen Interferenzen auftreten. Durch den schnellen Frequenzsprung ist es allerdings nicht möglich, die Medienbelegung vor dem Senden per Carrier Sense zu überprüfen. Damit Bluetooth andere Kommunikationssysteme, die auf dem 2,4 GHz Band senden ohne Frequency Hopping einzusetzen (wie beispielsweise IEEE 802.11), nicht zu sehr stört, wurde mit Bluetooth 1.2 das sogenannte *Adaptive Frequency Hopping* (AFH) eingeführt. Mit AFH nutzt Bluetooth von den möglichen 79 Frequenzen jene nicht, auf denen andere drahtlose Kommunikationssysteme erkannt wurden. Mindestens 20 Frequenzen werden allerdings immer verwendet, auch wenn auf diesen andere Kommunikationssysteme erkannt wurden.
Die Reichweite eines Bluetooth-Gerätes hängt davon ab, welcher Klasse es angehört [Blu13a]: Für den industriellen Einsatz ist die erste Klasse vorgesehen, sie erlaubt den Geräten mit bis zu 100 mW zu senden und so eine Reichweite von ca. 100 m zu erreichen. Die Geräte-Klasse 2 wird meistens in Mobiltelefonen verwendet, erlaubt eine Sendestärke von 2,5 mW und damit eine Reichweite von ca. 10 m. Geräte der Klasse 3 senden mit bis zu 1 mW und erreichen damit eine Reichweite von nur etwa einem Meter.
Die von Bluetooth definierten Topologien sind *Piconetze* und *Scatternetze*. In einem Piconetz sind mit einem Masterknoten bis zu sieben aktive Slaveknoten als Sterntopologie verbunden. Weitere Knoten können sich im Park-Modus befinden und bei Bedarf aufgeweckt werden. Die Kommunikation wird vom Masterknoten gesteuert, indem dieser Sendeslots an die Slaves vergibt und so Kollisionen verhindert. Die Frequenz-Sprungsequenz eines Piconetzes ergibt sich aus der Adresse des Masterknotens. Ein Knoten kann in mehreren Piconetzen angemeldet sein und bildet

damit ein Scatternetz. Er kann somit Routing-Funktionalität zwischen den Piconetzen übernehmen, auch wenn dies nicht durch die Bluetooth-Core-Spezifikation verlangt oder beschrieben wird.

Das Sicherheits-Modell von Bluetooth unterstützt mittlerweile vielfältige Sicherheitsfunktionen unter Einsatz unterschiedlicher kryptographischer Verfahren. Schon die erste Bluetooth-Version unterstütze Verschlüsselung basierend auf dem E0 Algorithmus, der auf Massey und Rueppel zurückgeht. Version 2.1 führt weitere kryptografische Verfahren ein, die auch die Integrität der Nachrichten garantieren können (u.a. SHA-256). In Bluetooth 4.0 Low Energy kommt der Advanced Encryption Standard (AES) im CCM-Modus zur Verschlüsselung zum Einsatz und die neuste Bluetooth-Version 4.1 [Blu13b] führt weitere Verfahren für sicheres Pairing (basierend auf P-256-Elliptic curve), Geräte-Authentifizierung (HMAC-SHA-256 und AES-CTR) sowie für die Sicherstellung der Nachrichten-Integrität (AES-CCM) ein.

Für den Einsatz im industriellen Produktionsumfeld galten die ersten Bluetooth-Versionen aus folgenden Gründen als nur sehr eingeschränkt geeignet (siehe [MTA05]):

1. Der Energieverbrauch durch Bluetooth ist recht hoch verglichen mit anderen Standards wie etwa IEEE 802.15.4 / ZigBee.
2. Piconetze bestehen aus einem Master und maximal sieben Slaves. Um größere Topologien umzusetzen müssen mehrere Piconetze zu Scatternetzen verknüpft werden.
3. Zwischen den Masterknoten zweier Piconetze darf nicht mehr als ein Hop liegen.
4. Schlafende Knoten benötigen mit typischerweise 3 Sekunden verhältnismäßig lange, um wieder aufzuwachen, wodurch Energieeinsparung zusätzlich erschwert wird.

Mit Version 4.0 wurde *Bluetooth Low Energy*, auch *Bluetooth Smart* genannt, eingeführt. Ziel der neuen Version soll vor allem sein, einen möglichst geringen Energieverbrauch von Bluetooth-Geräten zu erzielen und so mit Knopfzellenbatterien eine Lebensdauer von Jahren zu erreichen [Blu13c]. Dafür beträgt die maximale Übertragungsrate mit 1 Mbit/s weniger als bei Bluetooth 2.0 + EDR und Bluetooth 3.0 + HS. Als Anwendungsgebiete nennt die Bluetooth SIG die Bereiche Automobil, Wellness, Smart Energy, Unterhaltung, Heimautomatisierung, Sicherheit und Überwachung sowie Sport und Fitness. Aber auch für den Bereich der industriellen Produktion gewinnt Bluetooth mit Version 4.0 an Attraktivität: Mit einem deutlich geringeren Energieverbrauch und kürzeren Wakeup-Zeiten von unter 100 ms [PW10] gleicht Bluetooth Low Energy wichtige Nachteile aus. Allerdings bietet Bluetooth Low Energy keine Unterstützung für Multi-Hop-Topologien [GOP12] und ist damit für größere drahtlose Netzwerke ungeeignet. Darüber hinaus wurde Bluetooth 4.0 erst 2010 veröffentlicht und zwischenzeitlich hatte der Bedarf nach geeigneten drahtlosen Kommunikationssystemen im industriellen Bereich bereits zur Entwicklung spezialisierter Standards geführt, die 2007 (WirelessHART) bzw. 2009 (ISA100.11a) veröffentlicht wurden.

3.1.2 IEEE 802.11 WLAN

Der als WLAN bekannte internationale Standard IEEE 802.11 [IEE12a] wird hauptsächlich genutzt, um tragbare Geräte wie Laptops, Smartphones oder Tablets an breitbandige Internetverbindungen anzubinden. Der Standard ermöglicht schnelle drahtlose Übertragungen auf 2,4 GHz, 5 GHz oder neuerdings auch 60 GHz mit Datenraten von ursprünglich 2 Mbit/s bis hin zu mehreren Gbit/s (s. Tab. 3.2).

Mit handelsüblichen Geräten wird eine Reichweite von 30 bis 100 m erreicht, mit speziellen Antennen bei Sichtkontakt im Freien sind bis zu 300 m möglich. Damit hat WLAN typischerweise

Tabelle 3.2.: IEEE 802.11 Standards, Frequenzen und Datenraten

IEEE Standard	Frequenz (GHz)	Brutto-Datenrate (Mbit/s)	Veröffentlicht
802.11	2,4	2	1999
802.11a [IEE03a]	5	54	1999
802.11b [IEE99]	2,4	11	1999
802.11g [IEE03b]	2,4	54	2003
802.11n [IEE09a]	2,4 und 5	600	2009 (Entwurf 2006)
802.11ac [IEE13b]	5	1.300	vor. 2014
802.11ad [IEE12b]	60	7.000	2012

eine größere Reichweite als Bluetooth.
Der Standard sieht sowohl Infrastruktur-basierte Topologien vor, bei denen sich jeder Client mit einem Acces Point verbindet, als auch Ad-Hoc Topologien, bei denen sich zwei oder mehr Clients direkt miteinander verbinden. Darüber hinaus wurde mit IEEE 802.11s [IEE11b] Unterstützung für Mesh-Topologien standardisiert.
Der IEEE 802.11 Standard selbst spezifiziert nicht alle Ebenen eines Kommunikationsprotokolls, sondern lediglich Physical Layer und MAC Layer. Typischerweise kommt das Internet-Protokoll IP auf der Netzwerk-Ebene und TCP bzw. UDP auf Transport-Ebene zum Einsatz, um WLAN-Netzwerke an das Internet anzubinden.
Mit WPA (Wi-Fi Protected Access) und WPA2 können WLAN-Netze auf AES basierend sicher verschlüsselt werden. Darüber hinaus verwenden einige Anwendungen zusätzlich eine Ende-zu-Ende-Verschlüsselung mit SSL/TLS auf Transport-Ebene.

Die ursprüngliche WLAN-Spezifikation IEEE 802.11 sieht zwei unterschiedliche Möglichkeiten des Medien-Zugriffs vor:
Die erste Möglichkeit ist die *Distributed Coordination Function* (DCF), die auf *Carrier Sense Multiple Access with Collission Avoidance* (CSMA/CA) basiert. Bei dieser Form des Medienzugriffs wird vor dem Senden das Medium abgehört und der Sendevorgang nur gestartet, falls das Medium als frei erkannt wurde. Auf diese Weise kann die Kollisionswahrscheinlichkeit verringert, Kollisionen aber nicht verhindert werden. Insbesondere problematisch ist, dass die Wahrscheinlichkeit von Kollisionen mit der Anzahl der Knoten im Netz steigt. Darüber hinaus erlaubt CSMA/-CA keine ausreichende Dienstgüte-Unterstützung, da es unbegrenzt oft zu Kollisionen kommen kann und somit keinerlei Garantien für maximale Verzögerungen gegeben werden können. Aus diesen Gründen ist diese Art des Medien-Zugriffs für den Einsatz im Produktionsumfeld ungeeignet.
Die zweite Möglichkeit des Medien-Zugriffs bei IEEE 802.11 ist die *Point Coordination Function* (PCF). Bei dieser Möglichkeit wird der Medienzugriff durch den Access Point geregelt, daher ist sie im Ad-Hoc Modus ohne Access Point nicht verfügbar. Bei PCF teilt der Access Point die Zeit durch das regelmäßige Senden von Beacon Rahmen ein. Zwischen zwei Beacon Rahmen gibt es zwei Phasen, die Contention Free Period (CFP) und die Contention Period (CP). Während der CP wird der Medienzugriff wie bei DCF durch CSMA/CA geregelt. In der CFP sendet der Access Point Contention-Free-Poll (CF-Poll) Pakete an die einzelnen Stationen und erteilt ihnen so die Möglichkeit zum Senden. Da der Access Point immer nur einer Station ein CF-Poll Paket schickt, kommt es in der CFP nicht zu Kollisionen. Daher wäre diese Art des Medienzugriffs für den Einsatz im Produktionsumfeld prinzipiell geeignet. Der Access Point würde dann idealerweise nicht allen Knoten der Reihe nach CF-Poll Pakete schicken, sondern beim Polling nach

einem festgelegtem Schedule verfahren.
Mit IEEE 802.11e [IEE05a] wurde eine verbesserte Dienstgüte-Unterstützung auf Medienzugriffs-Ebene geschaffen. Im *Enhanced Distributed Channel Access* (EDCA) Modus wird grundsätzlich wie im DCF über CSMA/CA der Medienzugriff geregelt, allerdings können wichtige Pakete gegenüber unwichtigen priorisiert werden. Auch hier sind keinerlei Garantien für maximale Verzögerungen möglich. Im *HCF Controlled Chanel Access* (HCCA) wird ähnlich wie im PCF der Medienzugriff durch den Access Point geregelt, allerdings kann der Access Point bei Bedarf jederzeit eine CFP beginnen. Darüber hinaus werden Traffic Classes und Traffic Streams definiert, die es dem Access Point erlauben, die Polls nicht nur gleichmäßig auf die Stationen zu verteilen, sondern Prioritäten und vorhandene Datenströme zu berücksichtigen, etwa um Voice over IP (VoIP) über WLAN besser zu unterstützen. Auch HCCA wäre prinzipiell für ein drahtloses Kommunikationssystem im Produktionsbereich geeignet, da es kollisionsfreien Verkehr ermöglicht.

Neben der hohen Datenrate zählt auch die enorme Verbreitung von WLAN zu einer der wichtigsten Stärken. Sie dürfte dafür verantwortlich sein, dass Endgeräte für WLAN wie etwa sogenannte WLAN Sticks mit USB-Schnittstelle mittlerweile für unter 5 Euro angeboten werden. Darüber hinaus führt das starke Interesse an WLAN dazu, dass der IEEE 802.11 Standard stets aktualisiert und durch Erweiterungen ausgebaut wird. Die bekanntesten Aktualisierungen IEEE 802.11b, IEEE 802.11g und IEEE 802.11n haben höhere Datenraten ermöglicht. Dieser Trend wird von den jüngsten Aktualisierungen IEEE 802.11ac und IEEE 802.11ad fortgesetzt. Hinzu kommen andere Erweiterungen, wie etwa die Einführung von WPA2 Verschlüsselung (IEEE 802.11i [IEE04a]), Mesh-Netzwerke (IEEE 802.11s) oder bessere Dienstgüte-Unterstützung (IEEE 802.11e).

Trotz der vielen Vorteile von WLAN ist es für den Einsatz als drahtloses Kommunikationssystem in der Produktion aus mehreren Gründen nur stark eingeschränkt geeignet (vgl. [MTA05]): Das größte Problem ist, dass der Energieverbrauch von WLAN-Hardware zu hoch ist, um einen sinnvollen Batteriebetrieb zu ermöglichen. Damit nicht täglich Batterien gewechselt werden müssen, würde dies bedeuten, dass jeder Knoten an eine externe Stromversorgung angeschlossen werden müsste. Dadurch, dass zu allen Knoten eine Stromversorgung per Kabel verlegt werden muss, entfallen aber viele Vorteile der drahtlosen Kommunikation.
Darüber hinaus ist WLAN recht empfindlich gegen Interferenzen, da es keine Frequency Hopping Techniken nutzt. Solange Interferenzen auf dem durch WLAN genutzten Kanal auftreten, ist keine Übertragung möglich, was zu unbeschränkt langen Verzögerungen führen kann.

3.1.3 IEEE 802.15.4

Der internationale Standard IEEE 802.15.4 [IEE11a] spezifiziert Physical Layer und MAC Layer für *Wireless Personal Area Networks* (WPAN), also drahtlose Netzwerke mit einer geringeren Reichweite als bei WLAN-Netzen. Er bildet die Grundlage für eine Reihe weiterer Protokolle, darunter Zigbee (s. Kapitel 3.1.4), WirelessHART (s. Kapitel 3.1.6), ISA 100.11a (s. Kapitel 3.1.7) und WIA-PA (s. Kapitel 3.1.8), welche den Physical Layer und teilweise auch den MAC Layer von IEEE 802.15.4 übernehmen und die darüber liegenden Layer selbst spezifizieren.

Im Gegensatz zu IEEE 802.11 legt IEEE 802.15.4 den Fokus nicht auf hohe Datenraten und große Reichweiten, sondern auf geringen Energieverbrauch und geringe Komplexität, um auch kleine und günstige drahtlose Knoten ohne externe Energieversorgung und mit einfachen Prozessoren zu unterstützen. Dabei werden geringere Reichweiten und Datenraten explizit in Kauf genom-

men. Mit dieser Zielsetzung wird IEEE 802.15.4 den Anforderungen an drahtlose Kommunikationssysteme im Produktionsbereich (s. Kapitel 2) am besten gerecht, was erklärt, dass die speziell für solche Einsatzzwecke entworfenen Protokolle WirelessHART, ISA 100.11a und WIA-PA alle diesen Standard als Basis verwenden.

Der Physical Layer von IEEE 802.15.4 sieht unterschiedliche Übertragungsfrequenzen mit teilweise unterschiedlichen Modulationen vor, wodurch unterschiedliche Datenraten erreicht werden (s. Tab. 3.3). Nach Erscheinen des ursprünglichen Standards 2003 wurde der Physical Layer mehrfach um weitere mögliche Frequenzen erweitert. Durch die Vielzahl der möglichen Frequenzen ist es möglich, dass ein System die Frequenz nutzt, die den Anforderungen an Energieverbrauch, Datenrate und Reichweite am ehesten gerecht wird.

Tabelle 3.3.: IEEE 802.15.4 Standards, deren Frequenzen und Datenraten

IEEE Standard	Frequenz (MHz)	Modulation	Datenrate (kbit/s)	Kanäle
802.15.4-2003 [IEE03c]	868	BPSK	20	1
802.15.4-2003	915	BPSK	40	10
802.15.4-2003	DSSS 2.450	O-QPSK	250	16
802.15.4-2006 [IEE06]	868	ASK	250	1
802.15.4-2006	868	O-QPSK	100	1
802.15.4-2006	915	O-QPSK	250	10
802.15.4-2006	915	ASK	250	10
802.15.4a-2007 [IEE07]	CSS 2.450		250	14
802.15.4a-2007	CSS 2.450		1.000	14
802.15.4a-2007	UWB 250 - 750		850 - 27.000	1
802.15.4a-2007	UWB 3.244 - 4.742		850 - 27.000	4
802.15.4a-2007	UWB 5.944 - 10.234		850 - 27.000	11
802.15.4c-2009 [IEE09b]	780	O-QPSK	250	8
802.15.4c-2009	780	MPSK	250	8
802.15.4d-2009 [IEE09c]	950	GFSK	100	12
802.15.4d-2009	950	BPSK	20	10
802.15.4f-2012 [IEE12d]	433	MSK	31,25	15
802.15.4f-2012	433	MSK	100	15
802.15.4f-2012	433	MSK	250	15
802.15.4f-2012	2.450	MSK	250	42
802.15.4f-2012	LRP UWB	M. PPM	31,25	3
802.15.4f-2012	LRP UWB	OOK	250	3
802.15.4f-2012	LRP UWB	OOK	1000	3

Viele der vorgesehenen Frequenzen sind nur in einigen Ländern freigegeben. Die wohl am häufigsten verwendete Frequenz ist 2,4 GHz (DSSS) mit einer Datenrate von 250 kbit/s. Dieses auch von Bluetooth und WLAN verwendete ISM-Band hat den Vorteil, dass es weltweit freigegeben ist, allerdings auch den Nachteil, dass es deshalb von vielen anderen Systemen verwendet wird und die Wahrscheinlichkeit, dass Interferenzen auftreten, somit hoch ist. Auf der 2,4 GHz Frequenz werden im Innenbereich Reichweiten bis 30 m erreicht, im Außenbereich sind bis zu 70 m möglich [PRML06].

Eine Besonderheit von IEEE 802.15.4 ist, dass mit IEEE 802.15.4a-2007 der Physical Layer um optionale Untersützung für Ultra-Wideband (UWB) erweitert wurde. Mit dieser Übertragungs-

technik können bei geringer Sendeleistung auf einem sehr breiten Frequenzband Daten mit einer hohen Robustheit oder hohen Datenrate übertragen werden (Details s. Kapitel 3.1.5).
Die ursprüngliche IEEE 802.15.4 Spezifikation sieht kein Frequency Hopping wie bei Bluetooth vor. Ein IEEE 802.15.4 Transceiver bleibt also genau wie bei IEEE 802.11 stets auf der gleichen Frequenz und kann daher erheblich gestört werden, wenn auf dieser Frequenz häufig Interferenzen auftreten. Mit IEEE 802.15.4e [IEE12c] wurde 2012 der Standard um optionales *Channel Hopping* erweitert. Damit ist es möglich, dass IEEE 802.15.4 Geräte ähnlich wie Bluetooth-Geräte zwischen mehreren Kanälen wechseln und somit unempfindlicher gegen Störungen werden. Verglichen mit Bluetooth stehen aber deutlich weniger mögliche Kanäle zur Verfügung (je nach genutzter Frequenz und Modulation, s. Tab. 3.3) und das Wechseln zwischen den Frequenzen erfolgt nicht so schnell wie bei Bluetooth (je nach Konfiguration z.B. 100 mal pro Sekunde).

Der MAC Layer von IEEE 802.15.4 unterstützt durch CSMA/CA geregelten Wettbewerb und *Guaranteed Time Slots* (GTS), also exklusiv reservierte Zeitslots. Sogenannte durch *Beacon*-Frames abgetrennte *Superframes* beginnen mit einer Wettbewerbsphase, an die sich eine wettbewerbsfreie Phase mit GTS anschließen kann. Am Ende eines Superframes ist eine inaktive Phase möglich, in der kein Verkehr stattfindet und die Knoten in einen energiesparenden Schlafzustand wechseln können. Beacon-Frames werden vom Koordinator-Knoten gesendet, alternativ kann auf Beacon-Frames verzichtet werden und ausschließlich CSMA/CA basierter Wettbewerb genutzt werden.

Obwohl IEEE 802.15.4 nur Physical Layer und MAC Layer spezifiziert, also weder Multi-Hop-Übertragung noch Routing betrachtet, wird zwischen zwei Knotentypen unterschieden: Ein *Full Function Device* (FFD) kann als normaler Knoten oder Koordinator im Netz agieren und mit allen Knoten in Reichweite direkt kommunizieren. *Reduced Function Devices* (RFD) können dagegen nur mit einem FFD kommunizieren und nicht als Koordinator agieren. Somit können RFDs sehr einfache und günstiger Knoten (z.B. Sensorknoten) sein, die weniger Energie verbrauchen als FFDs. Neben Sterntopologien (RFDs und FFDs um einen FFD) sind auch Mesh-Netzwerke aus FFDs oder Stern-Mesh-Topologien aus vermaschten FFDs und damit verbundenen RFDs möglich. Wenn obere Layer eine Multi-Hop-Übertragungen ermöglichen, müssen RFDs keinerlei Routing-Funktionalität oder Relay-Aufgaben übernehmen, da sie ausschließlich mit einem FFD kommunizieren. Dies ermöglicht es RFDs, längere Zeit in einem energiesparenden Schlafmodus zu verbringen. Eines der FFDs im Netzwerk übernimmt die Rolle des *PAN Coordinators* und ist damit für Netwerk-Management-Aufgaben verantwortlich.

Der MAC Layer von IEEE 802.15.4 spezifiziert die Möglichkeit, die Kommunikation zwischen zwei Knoten mittels AES-128 symmetrisch zu verschlüsseln. Die Generierung der dafür benötigten Schlüssel wird vom Standard aber nicht genauer spezifiziert und wird somit den oberen Layern überlassen.

Wie bereits eingangs erwähnt, passt die Ausrichtung von IEEE 802.15.4 gut zu den Anforderungen an drahtlose Kommunikationssysteme im Produktionsbereich. Besonders der geringe Energiebedarf von IEEE 802.15.4 Knoten und die verhältnismäßig geringe Komplexität des Protokolls macht es für diesen Einsatzzweck geeignet. Dazu kommt auf MAC Ebene die Möglichkeit, Zeitslots exklusiv zu reservieren und somit Kollisionen zu vermeiden. Dazu ist die Unterscheidung von einfachen, günstigen und energiesparenden RFDs und komplexen FFDs hilfreich.
Obwohl IEEE 802.15.4 gut für den Einsatz im Produktionsumfeld geeignet ist, reicht es für sich genommen für diesen Einsatzzweck nicht aus. Da nur Physical Layer und MAC Layer spezifiziert werden, ist keine Multi-Hop-Übertragung möglich. Angesichts der geringen Reichweite von IEEE 802.15.4 kann in einer Produktionsanlage aber nicht mit einem Single-Hop-Netzwerk

gerechnet werden, daher ist ein zusätzliches Protokoll für die höheren Ebenen erforderlich.
Die ursprüngliche Spezifikation von IEEE 802.15.4 hat darüber hinaus die Schwäche, dass keine Frequency Hopping Techniken vorgesehen sind und durch Interferenzen unbegrenzt lange Störungen auftreten können. Diese Schwäche wurde allerdings mit der Einführung von Channel Hopping in IEEE 802.15.4e im Jahr 2012 beseitigt.

3.1.4 ZigBee

ZigBee [Zig08] ist ein internationaler Standard basierend auf IEEE 802.15.4. Er ist ausgerichtet auf günstige Knoten mit geringem Energiebedarf. Als mögliche Einsatzgebiete nennt der Standard Unterhaltungselektronik, Heim- und Gebäude-Automatisierung, industrielle Steuerung, PC-Peripherie, medizinische Sensor-Anwendungen, Spielzeug und elektronische Spiele.
Die ZigBee Spezifikation übernimmt sowohl den Physical Layer als auch den MAC Layer aus IEEE 802.15.4-2003. Dies bedeutet, dass ZigBee-Netzwerke entweder auf 868 MHz, 915 MHz oder 2,4 GHz arbeiten können und dort 20 kbit/s, 40 kbit/s respektive 250 kbit/s Datenrate erreichen können (s. Tab. 3.3). Da in dieser Version von IEEE 802.15.4 noch kein Channel Hopping vorgesehen ist und ZigBee kein eigenes Chanell Hopping umsetzt, sind ZigBee Netze recht empfindlich gegen Interferenzen.
Über die durch IEEE 802.15.4 spezifizierten Layer fügt Zigbee einen Network Layer an, der Stern-, Baum- und Mesh-Topologien unterstützt. Er bietet Routing-Funktionalität durch Einsatz des *Ad-hoc on Demand Distance Vector* (AODV) Algorithmus. Dadurch werden Routen von den Knoten selbst bestimmt und müssen nicht statisch konfiguriert sein. Bei AODV handelt es sich um ein reaktives Routingverfahren, d.h. in dem Moment, in dem ein Paket zu einem Zielknoten geschickt werden soll, zu dem keine Route bekannt ist, wird begonnen, die Route zu bestimmen.
ZigBee unterscheidet zwischen drei Knotentypen: Der *ZigBee Coordinator* entspricht dem PAN Coordinator des IEEE 802.15.4-Netzwerks. Ein *ZigBee Router* ist ein IEEE 802.15.4 FFD, der nicht der ZigBee Coordinator ist und Nachrichten zwischen Geräten routen kann. Ein *ZigBee end device* ist ein IEEE 802.15.4 RFD oder FFD, das weder der ZigBee Koordinator noch ein ZigBee Router ist.
Prinzipiell können ZigBee-Netzwerke genau wie IEEE 802.15.4-Netzwerke optional eine Synchronisation mit Beacons einsetzen. Allerdings sind Beacons nur in Stern- und Baum-Topologien erlaubt, nicht aber in Mesh-Topologien. Darüber hinaus müssen in Baum-Topologien die aktiven Perioden der Knoten, die Beacons senden, überlappungsfrei mit denen aller Nachbarn und deren Elternknoten sein.
Die ZigBee Architektur sieht diverse Sicherheits-Dienste für Schlüssel-Erzeugung, Schlüssel-Transport, Verschlüsselung und Geräte-Verwaltung vor. Die im IEEE 802.15.4 MAC Layer enthaltenen Sicherheitsfunktionen, wie etwa Verschlüsselung auf MAC-Ebene, werden vom ZigBee Standard nicht explizit erwähnt, daher könnte deren Nutzung Kompatibilitätsprobleme mit anderen Geräten hervorrufen [LSH08]. Allerdings spezifiziert ZigBee selbst Sicherheitsdienste, u.a. Verschlüsselung auf dem Network- und Appllication Layer, und nutzt dabei die vorhandenen Sicherheits-Mechanismen von IEEE 802.15.4. So wird z.B. der AES-CCM-Modus aus IEEE 802.15.4 erweitert, sodass er Unterstützung für Authentifizierung, Verschlüsselung oder beides bietet. Unicast-Nachrichten werden mithilfe eines Link-Keys verschlüsselt, einem Schlüssel den, nur Sender und Empfänger teilen. Broadcast-Nachrichten dagegen werden mit einem Network-Key geschützt, der allen Geräten im Netzwerk bekannt ist.
Da ZigBee auch in Hinblick auf den industriellen Einsatz konzipiert wurde, eignet es sich für den Einsatz in Produktionsanlagen für mehr Einsatzzwecke als Bluetooth oder WLAN (vgl.

[MTA05], [Bak05]). Insbesondere auf Grund des geringeren Energiebedarfs eignet es sich besser für batteriebetriebene Sensorknoten. Darüber hinaus ist die Unterstützung von Multi-Hop-Topologien inkl. Routing besser. Die umfangreichen Sicherheits-Dienste von ZigBee sind ebenfalls gut geeignet für den Einsatz im industriellen Umfeld.
Trotzdem weist ZigBee einige Schwächen auf, die es für den Einsatz als Kommunikationssystem im Produktionsumfeld nur bedingt geeignet erscheinen lassen. Lenvall et al. [LSH08] kritisieren vor allem, dass die Robustheit von ZigBee dem Einsatz im industriellen Umfeld nicht gewachsen sei. Dies wird zum einen darauf zurückgeführt, dass ZigBee kein Frequency oder Channel Hopping unterstützt und daher sehr anfällig für Störungen ist, insbesondere auch dann, wenn andere Systeme wie WLAN oder Bluetooth im gleichen Frequenzbereich eingesetzt werden. Allerdings wurde 2012 der ZigBee zugrunde liegende IEEE 802.15.4 Standard um Channel Hopping erweitert, sodass davon auszugehen ist, dass zukünftige ZigBee-Spezifikationen von dieser Möglichkeit Gebrauch machen werden und ZigBee damit robuster werden wird. Darüber hinaus kritisieren Lenvall et al., dass das Finden neuer Routen mit AODV, z.B. im Falle eines gebrochenem Links, einen sehr hohen Overhead erzeugt und zu erheblichen Verzögerungen führt. Dieses Problem könnte minimiert werden, indem beispielsweise im Falle eines gebrochenen Links eine bereits bekannte alternative Route genutzt wird. Diese Möglichkeit sieht der aktuelle ZigBee Standard allerdings nicht vor. Auf Grund der geringen Robustheit ist ZigBee für den Einsatz in Steuerung und Regelung laut Lenvall et al. nur eingeschränkt geeignet. Darüber hinaus sei der Batteriebetrieb von ZigBee Routern unrealistisch, da der Einsatz von CSMA/CA dazu führt, dass der Empfänger über einen großen Teil der Zeit empfangsbereit sein muss.

3.1.5 UWB

Mit Ultra Wide Band (UWB) bezeichnet man eine Funk-Technologie, bei der im Gegensatz zu konventionellen Funksystemen auf einem sehr breiten Frequenzband gesendet wird, allerdings mit einer sehr geringen Sendestärke. Das Leistungsdichtespektrum bleibt dabei unter der von elektronischen Gebrauchsgeräten erzeugten Rauschleistungsdichte [Eis06], sodass konventionelle schmalbandige Funksysteme nicht gestört werden, selbst wenn sie innerhalb des von UWB genutzten Frequenzbandes liegen. Aus diesem Grund können UWB-Systeme Frequenzen mitnutzen, die für andere Einsatzzwecke exklusiv lizensiert sind, ohne dass sie diese stören. Darüber ergibt sich durch das extrem breite Frequenzband, auf dem gesendet wird, die Möglichkeit, entweder eine sehr hohe Datenrate oder eine erhöhte Robustheit bei der Übertragung zu erreichen.

Zusammenfassend verspricht die UWB Technologie folgende Vorteile:

- Hohe Datenraten oder robuste Übertragung
- Geringer Energieverbrauch durch geringe Sendeleistung
- Keine Lizenzgebühren für Frequenzen
- Effektive Nutzung des verfügbaren Frequenzraumes

Nachteilig ist, dass auf Grund der geringen Sendeleistung im Vergleich zu schmalbandigen Verfahren nur geringe Reichweiten möglich sind und UWB sich daher nur für den Einsatz in WPANs eignet. Da sich mit UWB eine robuste Übertragung bei geringem Energieverbrauch realisieren lässt, ohne Lizenzgebühren für Frequenzen zahlen zu müssen, wird diese Technologie als besonders geeignet für den industriellen Einsatz angesehen [HA06, MTA05]. Damit die UWB Technologie diesen Markt erschließen kann, wird allerdings ein einheitlicher Standard benötigt.

Im Gegensatz zu dem bisher betrachteten Standards handelt es sich bei UWB nicht um einen einzelnen Standard zur drahtlosen Kommunikation, sondern um eine Technologie, die in unterschiedlichen Standards Verwendung finden kann. Da bei UWB sich in erster Linie die physikalische Datenübertragung von herkömmlichen schmalbandigen Systemen unterscheidet, stellt UWB zunächst eine alternative Art und Weise dar, den Physical Layer eines drahtlosen Kommunikationssystems zu realisieren. Es ist also naheliegend, den Physical Layer eines bisherigen Standards durch einen auf UWB Technologie basierenden zu ersetzen bzw. zu erweitern. Mit IEEE 802.15.4a wurde 2007 genau dies getan, nämlich der IEEE 802.15.4 Standard (s. Kapitel 3.1.3) um alternative physikalische Layer erweitert, die auf UWB basieren (s. Tab. 3.3). Ähnliche Bestrebungen, UWB in Bluetooth 3.0 zu verwenden, wurden 2009 zu Gunsten der Verwendung eines auf IEEE 802.11 basierenden High-Speed-Kanals auf Grund von Streitigkeiten um Lizenzgebühren aufgegeben [Hei09]. Zuvor hatten zwei Industrie-Organisationen, das UWB Forum und die WiMedia Alliance, angestrebt, einen auf UWB basierenden WPAN Standard für hohe Datenraten als IEEE 802.15.3a [IEE04b] zu standardisieren. Dabei favorisierte das UWB Forum mit Direct sequence-UWB (DS-UWB) eine andere Technologie als die WiMedia Alliance, welche Multiband Orthogonal Frequency Division Multiplexing (MB-OFDM) UWB favorisierte. Da beide Lager zu keiner Einigung kamen, wurde die Standardisierung von IEEE 802.15.3a im Januar 2006 eingestellt [UWB06]. Allerdings wurde die Spezifikation der WiMedia Alliance als ISO/IEC 26907 (Physical Layer) [Int07a] und ISO/IEC 26908 (MAC Layer) [Int07b] im Jahr 2007 international standardisiert.
Für den Einsatz als drahtloses Kommunikationssystem im Produktionsumfeld ist insbesondere die Erweiterung des IEEE 802.15.4 Standards um UWB interessant, da diese im Gegensatz zu den meisten anderen Bestrebungen, UWB zu standardisieren, nicht in erster Linie eine besonders hohe Datenrate, sondern eine robuste Übertragung bei geringem Energieverbrauch zum Ziel hat. Somit könnte UWB in drahtlosen Kommunikationssystemen im Produktionsbereich als Alternative oder in Kombination mit Channel Hopping eingesetzt werden, um die Robustheit dieser Systeme zu verbessern. Der erste IEEE 802.15.4a kompatible Transmitter wurde noch im Jahr 2007 von IMEC hergestellt [IME07], einen besonders energiesparenden Sensorchip auf Basis von IEEE 802.15.4a veröffentlichte Decawave 2011 [Ene11]. Insgesamt scheint die Marktdurchdringung von IEEE 802.15.4a allerdings noch äußert gering.

3.1.6 WirelessHART

Als WirelessHART wird die Erweiterung des HART Communication Protocol [HAR14a] (Highway Addressable Remote Transducer - HART) um drahtlose Kommunikation bezeichnet. HART ist ein Mitte der 80er Jahre von Rosemount Inc. entwickeltes Protokoll für die drahtgebundene Vernetzung von Geräten zur Steuerung, Regelung und Überwachung. Das zunächst proprietäre Protokoll wurde früh offen gelegt, ist mittlerweile als IEC 61158 [IEC14] zum internationalen Standard geworden und wird heute von der HART Communication Foundation (HCF) vermarktet und weiterentwickelt. Die Veröffentlichung von WirelessHART im Rahmen der HART 7 Spezifikation im Jahr 2007 geschah gegen den Widerstand von Honeywell, Mitglied des HFC-Rates [Ive07]. Das Unternehmen hatte kurz zuvor in einem offenen Brief an die Presse argumentiert, es bestünde kein Bedarf nach einem nur das HART-Protokoll unterstützenden drahtlosen Protokoll mehr, sobald das sich zu dieser Zeit noch in Entwicklung befindliche ISA 100.11a verfügbar wäre, auf dessen Basis unterschiedliche Protokolle, wie HART, Profibus, CIP (Common Industrial Protocol) und Foundation Fieldbus (über Tunnelling) verwendet werden können [Ive07]. Im April 2010 wurde WirelessHART als IEC 62591[IEC10] zum ersten internationalen

Standard für drahtlose Kommunikation in der Prozessautomatisierung [HAR10]. Aktuell listet die HFC auf Ihrer Website ca. 50 kompatible WirelessHART-Geräte sowie ca. 1.300 drahtgebundene HART-Geräte [HAR14b].

Der Protokoll-Stack von WirelessHART ist zwar abwärtskompatibel zu HART, unterstützt also HART-Befehle auf Anwendungsebene, die anderen Protokollebenen unterscheiden sich aber von HART. Auf dem Physical Layer kommt der IEEE 802.15.4-2006 Standard (s. Kapitel 3.1.3) zum Einsatz, wobei ausschließlich die Frequenz 2.450 MHz mit DSSS und O-QPSK Modulation und nur die weltweit zugelassenen Kanäle 11-25 verwendet werden. Damit erreichen WirelessHART-Netzwerke eine Brutto-Datenrate von bis zu 250 kbit/s.

Der MAC Layer von IEEE 802.15.4 wird von WirelessHART nicht übernommen, sondern durch einen eigenen ersetzt. Dabei wird ebenfalls TDMA verwendet, allerdings gibt es keine Beacons und keine CP. Stattdessen wird die Zeit basierend auf einer netzweiten Zeitsynchronisation vollständig in 10 ms Slots unterteilt. Durch die Definition von Superframes werden eine bestimmte Anzahl Zeitslots zusammengefasst, um wiederkehrende Slot-Zuordnungen zu ermöglichen. Durch einen zentralen Network Manager können die Slots eines Superframes Knoten zum Senden und Empfangen zugeordnet werden. Dabei können Slots exklusiv zugeordnet werden oder mehrere Knoten konkurrieren um einen Slot. Dadurch, dass exklusive Reservierungen möglich sind, kann ein zuverlässiger Medienzugriff ohne Kollisionen sichergestellt werden, was für den Betrieb in Produktionsanlagen entscheiden ist.
Ebenfalls um eine zuverlässigere Übertragung zu erzielen definiert WirelessHART ein Channel Hopping auf MAC-Ebene (ähnlich dem Channel Hopping in IEEE 802.15.4e). Ein Zeitslot eines Superframes wird dadurch immer wieder einem anderen Kanal zugeordnet, sodass Störungen auf einem Kanal nicht wiederholt die selbe Übertragung stören. Indem im gleichen Zeitslot auf unterschiedlichen Kanälen gleichzeitig kommuniziert wird, kann außerdem die Gesamtdatenrate des Netzwerks erhöht werden. Kanäle, die durch andere Systeme wie WLAN gestört werden, können durch eine *Channel Blacklist* manuell durch den Netzwerk-Administrator vom Channel Hopping ausgeschlossen werden.

Im Gegensatz zu ZigBee und dem IEEE 802.15.4 MAC Layer kennt WirelessHART keine Knotentypen mit reduziertem Funktionsumfang, wie RFDs bei IEEE 802.15.4 bzw. ZigBee end devices. Bei WirelessHART müssen alle Knoten volle Routing-Funktionalität bereitstellen. Damit sind WirelessHART-Netze stets Mesh-Netze, eine Beschränkung auf eine Stern- oder Stern-Mesh-Topologie ist nicht möglich. Durch die stark vermaschte Netzstruktur sollen WirelessHART-Netze besonders zuverlässig sein, denn wenn beispielsweise ein Knoten ausfällt, können alternative Routen gewählt werden. Um dies effizient zu ermöglichen, wird das sogenannte *Graph-Routing* eingesetzt, wobei eine berechnete Route nicht nur einen Pfad enthält, sondern eine Liste alternativer Pfade.

Die Anwendungsebene von WirelessHART übernimmt das HART-Protokoll wie es auch in drahtgebundenen HART-Systemen zum Einsatz kommt. Es basiert auf Kommandos, welche teilweise von allen HART kompatiblen Geräten unterstützt werden müssen, teilweise auf bestimmte Geräte-Klassen zugeschnitten oder auch gerätespezifisch sein können.

Eine detailliertere Betrachtung von WirelessHART im Vergleich mit ISA 100.11a erfolgt in Kapitel 3.2.

3.1.7 ISA 100.11a

Die International Society of Automation (ISA) [ISA] ist eine ursprünglich amerikanische, nun internationale, Organisation mit aktuell über 30.000 Mitgliedern weltweit[1], die Standards, Zertifizierungen, Ausbildungen und Publikationen rund um das Thema Automatisierungstechnik anbietet [ISA14a]. Im Jahr 2005 gründete ISA das ISA 100 Komitee mit dem Ziel, Standards im Bereich drahtloser Systeme für die Automatisierungs- und Regelungstechnik zu schaffen. Das Komitee besteht aus über 400 Experten im Bereich Automatisierung aus über 250 Firmen, wobei Endanwender explizit eingeschlossen wurden und deren Interessen im Vordergrund stehen sollen [ISA08]. Der erste und bisher wichtigste vom ISA 100 Komitee im Jahr 2009 verabschiedete Standard ist ISA 100.11a. Er definiert ein flexibles, drahtloses Kommunikationsprotokoll für industrielle Automatisierungssysteme und wurde 2012 als IEC 62734 [IEC12] genau wie WirelessHART zu einem offenen internationalen IEC-Standard. Eine weitere Bemühung des ISA 100 Komitees betraf die Koexistenz von WirelessHART und ISA 100.11a Systemen bzw. die Annäherung beider Standards (ISA 100.12 Sub-Komitee), welche allerdings nicht zu einer Einigung führte [IEE13a].
Aktuell führt die Website des ISA 100 Wireless Compliance Institute [ISA14c], einer Organisation des von ISA gegründeten Automation Standards Compliance Institutes (ASCI) [ISA14b], knapp 30 ISA 100.11a kompatible Geräte [ISA14d].

Genau wie WirelessHART und ZigBee übernimmt ISA 100.11a den Physical Layer von IEEE 802.15.4 und nutzt dabei genau wie WirelessHART lediglich die 2.450 MHz Frequenz (DSSS). Die maximal mögliche Datenrate beträgt damit ebenfalls 250 kbit/s, die Reichweite im Innenbereich ca. 30 m und im Außenbereich 70 m.

Der MAC Layer von ISA 100.11a übernimmt Teile des IEEE 802.15.4 MAC Layers für das Senden und Empfangen von Rahmen. Allerdings werden viele Funktionen des IEEE 802.15.4 MAC Layers nicht verwendet. Beispielsweise verbinden sich ISA 100.11a Geräte nie mit einem Koordinator im Sinne von IEEE 802.15.4 und sämtliche Funktionen, die FFDs betreffen, werden nicht genutzt. Stattdessen wird der IEEE 802.15.4 MAC Layer mit einer MAC-Erweiterung um diverse Funktionen ergänzt. Beispielsweise werden Acknowledgements um Zeitinformation ergänzt, um eine Uhrenkorrektur durchzuführen. Allerdings wird dazu nicht der IEEE 802.15.4 MAC Layer verändert, sondern es werden Acknowledgements in der MAC-Erweiterung umgesetzt und als IEEE 802.15.4 Datenrahmen gesendet. Die Acknowledgement-Funktion von IEEE 802.15.4 bleibt also unbenutzt. Ebenso wird die Backoff-Funktionalität zur Neuübertragung aus IEEE 802.15.4 nicht verwendet, sondern eine erweiterte Methode zur Neuübertragung eingesetzt. Diese ermöglicht es nicht nur, den Rahmen mit einer zeitlichen Verzögerung zu senden, sondern ggf. auch zu einem anderen Knoten oder auf einem anderen Kanal.
Genau wie WirelessHART implementiert auch ISA 100.11a ein Channel Hopping auf MAC-Ebene und kombiniert so TDMA mit FDMA. Dabei erlaubt es anders als WirelessHART allerdings die Nutzung aller 16 Kanäle, die IEEE 802.15.4 auf 2,4 GHz spezifiziert. Neben manuell konfigurierten Channel Blacklists, wie sie WirelessHART kennt, setzt ISA 100.11a optional *Adaptive Channel Blacklisting* ein, bei dem jedes Gerät autonom problematische Kanäle blacklisten kann.
Der TDMA-Mechanismus von ISA 100.11a teilt die Zeit genau wie WirelessHART in Superframes auf, welche aus exklusiv reservierten Zeitslots bestehen, in denen jeweils eine Übertragung stattfinden kann. Anders als WirelessHART spezifiziert ISA 100.11a jedoch keine feste Länge ei-

[1] Im Rahmen dieser Masterarbeit wurde ich selbst studentisches Mitglied von ISA.

nes Zeitslots, sondern erlaubt es, die Zeitslotdauer (für alle Zeitslots) zu konfigurieren. Eine Erhöhung der Zeitslotdauer ist vor allem dann nötig, wenn sogenannte Dualcast-Übertragungen genutzt werden sollen: Bei einer solchen Übertragung wird ein Rahmen an zwei Empfängerknoten gleichzeitig übertragen, und beide Knoten bestätigen nacheinander den Empfang mit einem Acknowledgement. Da hierfür ein 10 ms Zeitslot nicht ausreicht, muss eine längere Zeitslotdauer gewählt werden, wenn diese Funktion genutzt werden soll.
Zusätzlich zu exklusiv reservierten Zeitslots, die eine wettbewerbsfreie Übertragung ermöglichen, bietet ISA 100.11a mit *Slow Hopping Perioden* auch die Möglichkeit, wettbewerbsbasierten Zugriff zu erlauben. In einer Slow Hopping Periode werden die Kanäle nicht so schnell gewechselt wie sonst, damit mittels CSMA/CA wettbewerbsbasierter Zugriff möglich wird. Welche Knoten am Wettbewerb in einer Slow Hopping Periode teilnehmen, kann konfiguriert werden. Seltene ereignisbasierte Nachrichten sollen vom Slow Hopping Mechanismus profitieren, ohne dass zu viele exklusive Zeitslots für sie konfiguriert werden müssen. Da allerdings wettbewerbsbasierter Zugriff keine zuverlässigen Übertragungen garantiert, ist er nur für Datenströme mit einer Best-Effort Dienstgüte geeignet oder kann zusätzlich zu exklusiven Reservierungen eingesetzt werden, um die Dienstgüte über die garantierte Basisdienstgüte hinaus zu steigern.

Obwohl ISA 100.11a die FFD-Funktionen aus dem IEEE 802.15.4 MAC Layer nicht nutzt, unterscheidet es ähnlich wie ZigBee und IEEE 802.15.4 zwischen Knoten mit eingeschränktem Funktionsumfang und Knoten mit weiteren Funktionen: Sogenannte *I/O Devices* sind i.d.R. Aktuator- oder Sensorknoten mit eingeschränktem Funktionsumfang, die insbesondere keine Routing-Funktionalität übernehmen. Diese wird von *Routern* übernommen. Daneben gibt es Gateways, System Manager, Security Manager und Backbone Router. Dabei werden Knoten nicht in Typen eingeteilt, sondern Knoten übernehmen eine oder mehrere dieser Rollen: So kann ein Knoten Router und I/O Device sein oder nur eines davon.
Das Routing von ISA 100.11a kennt wie WirelessHART mit *Graph-Routing* ein Routing-Verfahren, bei dem alternative Pfade vordefiniert sind, auf die im Fehlerfall ausgewichen werden kann. So soll die Redundanz erhöht und die Zuverlässigkeit des Systems verbessert werden.

Auf Anwendungsebene spezifiziert ISA 100.11a ein flexibles objektorientiertes Protokoll, erlaubt aber auch die Tunnelung von anderen Protokollen wie beispielsweise dem HART-Protokoll.

Das ISA 100.11a Protokoll wird im Zuge des Vergleichs mit WirelessHART in Kapitel 3.2 detaillierter betrachtet werden.

3.1.8 WIA-PA

WIA-PA (Wireless Network for Industrial Automation - Process Automation) ist ähnlich wie WirelessHART und ISA 100.11a ein Standard für drahtlose Kommunikationssysteme zur Prozess-Automatisierung im Produktionsbereich. Der Standard wurde seit 2007 von der Chinese Industrial Wireless Alliance unter der Federführung des Shenyang Institute of Automation der Chinesischen Akademie der Wissenschaften entwickelt [LLYL13]. Der Standard wurde 2009 als internationaler Standard IEC 62601 [IEC11] angenommen. Da trotzdem recht wenige internationale Publikationen zu WIA-PA gefunden werden konnten, wird die aktuelle Bedeutung des Standards für den internationalen Markt als gering eingeschätzt. Aus diesem Grund wird im Rahmen dieser Arbeit nur ein grober Überblick gegeben und kein detaillierter Vergleich mit WirelessHART und ISA 100.11a durchgeführt.

Als Physical Layer verwendet WIA-PA genau wie WirelessHART und ISA 100.11a die 2,4 GHz DSSS-Kanäle des IEEE 802.15.4 Standards. Der MAC Layer basiert wie der von ISA 100.11a

ebenfalls auf IEEE 802.15.4. Um eine hohe Zuverlässigkeit zu erreichen, obwohl andere Systeme das gleiche Band nutzen, setzt WIA-PA unterschiedliche Frequency Hopping Mechanismen auf MAC-Ebene ein [LLYL13]: Beim *Adaptive Frequency Hopping* wir der Kanal nur dann gewechselt, wenn schlechte Bedingungen auf dem aktuell genutzten Kanal erkannt werden. Beim *Timeslot Hopping* wird dagegen wie bei WirelessHART und ISA 100.11a der Kanal bei jedem neuen Zeitslot gewechselt, unabhängig davon, ob der bisherige Kanal gestört wurde oder nicht.
Der TDMA-Mechanismus von WIA-PA basiert stark auf IEEE 802.15.4: Er trennt Superframes mit Beacons ab und erlaubt sowohl wettbewerbsbasierten Zugriff in der *Contention Access Period* als auch exklusiven Zugriff in der *Contention Free Period* [ZLY09].

Ähnlich wie ISA 100.11a kennt WIA-PA unterschiedliche Knotentypen und unterscheidet zwischen Routing-Knoten und Field-Devices, die nicht routen können. Damit bildet ein WIA-PA-Netzwerk i.d.R. eine Stern-Mesh-Topologie. Das Routing verspricht mit *redundanten Pfaden* ähnlich wie das Graph-Routing in WirelessHART und ISA 100.11a eine zuverlässigere Übertragung.

3.1.9 Fazit

Die nicht speziell für den Einsatz zur Automatisierung im Produktionsbereich konzipierten drahtlosen Kommunikationssysteme weisen bedeutende Schwächen auf und sind daher nur sehr eingeschränkt für diese Anwendung geeignet. Beim klassischen Bluetooth und bei WLAN ist der Energiebedarf zu hoch, um batteriebetriebene Knoten mit einer praktikablen Akkulebensdauer zu ermöglichen. Bluetooth Low Energy ermöglicht zwar sinnvollen Batteriebetrieb, allerdings keine Multi-Hop-Topologien, womit auf Grund der geringen Reichweiten nur kleine Netzwerke möglich sind. IEEE 802.15.4 ist eine geeignete Basistechnologie für den Einsatzbereich, spezifiziert aber Netzwerk- und Anwendungsebene nicht und ist daher ohne ein darauf aufbauendes Protokoll unzureichend. Außerdem fehlte lange Zeit eine Unterstützung für Channel Hopping auf MAC-Ebene. Diese Schwäche überträgt sich auf ZigBee, welches die IEEE 802.15.4-2003 MAC-Ebene, welche noch kein Channel Hopping enthält, übernimmt, ohne ein eigenes Channel Hopping hinzuzufügen. Außerdem ist das reaktive Routingverfahren von ZigBee für den Einsatz in Netzwerken im Produktionsbereich nicht optimal. UWB verspricht als Basistechnologie für den Einsatz im Produktionsbereich wichtige Vorteile. Allerdings ist die Marktdurchdringung noch zu gering und es fehlen Protokolle auf höherer Ebene, welche aus den Vorteilen von UWB einen Nutzen ziehen.

WirelessHART stellt als erstes Protokoll speziell für drahtlose Kommunikationssysteme im Produktionsbereich einen bedeutenden Fortschritt gegenüber den unspezifischeren Protokollen dar. Es ermöglicht eine zuverlässige Übertragung in Multi-Hop-Netzwerken und batteriebetriebene Knoten. Kompatible Hardware ist verfügbar und zeigt, dass WirelessHART praxistauglich ist. ISA 100.11a bietet ebenfalls zuverlässige Übertragung in Multi-Hop-Netzwerken mit Unterstützung für batteriebetriebene Knoten. Durch die Unterscheidung von unterschiedlichen Knoten-Rollen erlaubt es ISA 100.11a im Gegensatz zu WirelessHART, einfachere Sensor- und Aktuatorknoten ohne Routing-Funktionalität einzusetzen und so zusätzlich Energie einzusparen. Insgesamt ist ISA 100.11a flexibler als WirelessHART, wie sich im detaillierten Vergleich in Kapitel 3.2 noch genauer herausstellen wird. Der chinesische Standard WIA-PA bietet ähnliche Funktionalitäten wie WirelessHART und ISA 100.11a, kann allerdings auf Grund unzureichender Informationen hier nicht näher betrachtet werden.

3.2 Vergleich von WirelessHART und ISA 100.11a

Nachdem bereits ein Überblick über WirelessHART und ISA 100.11a gegeben wurde, sollen die beiden Standards nun im Detail verglichen werden. Dabei werden die einzelnen Ebenen der Protokolle vom Physical Layer bis zum Application Layer untersucht und jeweils die Gemeinsamkeiten und Unterschiede herausgestellt.

3.2.1 Physical Layer

Auf dem Physical Layer übernehmen beide Standards den Physical Layer von IEEE 802.15.4 mit kleinen Änderungen. Dabei beschränken sie sich auf das 2,45 GHz Band mit DSSS und Q-PSK-Modulation, die eine Brutto-Datenrate von 250 kbit/s und 16 überlappungsfreie Kanäle (11-26) bietet (s. Tab. 3.3). Die von IEEE 802.15.4-2006 spezifizierten 868 MHz bzw. 915 MHz Frequenzbänder sowie der später hinzugefügte UWB-basierte Physical Layer können in beiden Standards bisher nicht genutzt werden. Da nur die Kanäle 11-25 weltweit frei genutzt werden dürfen, verbietet WirelessHART generell die Nutzung von Kanal 26, ISA 100.11a dagegen erlaubt die Nutzung optional. Darüber hinaus ergänzen die Standards einige Anforderungen an den Physical Layer und schränken teilweise Funktionen ein. Beispielsweise fordert ISA 100.11a vom Transceiver, dass er in weniger als 200 µs den Kanal wechseln kann. Da der ursprüngliche IEEE 802.15.4 Standard im Gegensatz zu ISA 100.11a und WirelessHART keinen Kanalwechsel vorgesehen hatte, wird diese Anforderung zusätzlich formuliert. Außerdem wird beispielsweise verlangt, dass der CSMA Mechanismus optional ist und von den oberen Ebenen deaktiviert werden kann. Dies wird genutzt, um die Übertragung in exklusiv reservierten Slots direkt beginnen zu können, ohne zunächst das Medium abhören zu müssen.

Für künftige Versionen des ISA 100.11a Standards wird explizit die Möglichkeit erwähnt, dass weitere Physical Layer hinzugefügt werden.

3.2.2 MAC Layer

Wie bereits erwähnt, übernimmt ISA 100.11a Teile des IEEE 802.15.4 MAC Layers, wohingegen WirelessHART einen komplett eigenen MAC Layer spezifiziert. Allerdings nutzt ISA 100.11a nur die grundlegenden MAC-Funktionalitäten zum Senden eines Rahmens. Nicht einmal einfache Funktionen, wie das Bestätigen eines erfolgreichen Empfangs per Acknowledgement, wird aus dem IEEE 802.15.4 Standard übernommen. Stattdessen wird ein eigener Acknowledgement-Mechanismus unter Verwendung von normalen IEEE 802.15.4 Datenrahmen umgesetzt. Mit den MAC-Erweiterungen, die ISA 100.11a einführt, um den IEEE 802.15.4 MAC Layer zu ergänzen, ähneln sich die Funktionsweisen der MAC Layer von WirelessHART und ISA 100.11a stark.

Der Medienzugriff wird bei beiden Standards durch TDMA kombiniert mit FDMA geregelt, wie es in Abb. 3.1 grafisch veranschaulicht wird. Die Zeit wird dabei in *Superframes* aufgeteilt, die periodisch wiederkehren. Superframes sind ihrerseits in *Zeitslots* aufgeteilt, wobei die 15 (WirelessHART) bzw. optional 16 (ISA 100.11a) Kanäle parallel in Zeitslots aufgeteilt werden. Außerdem können Knoten, die weit genug voneinander entfernt sind, im selben Zeitslot den selben Kanal nutzen (SDMA).

In jedem Zeitslot kann auf jedem Kanal maximal eine Übertragung inkl. Acknowledgement stattfinden. Die zeitliche Aufteilung eines Zeitslots in Übertragung des eigentlichen Rahmens und des Acknowledgements wird in Abb. 3.2 veranschaulicht.

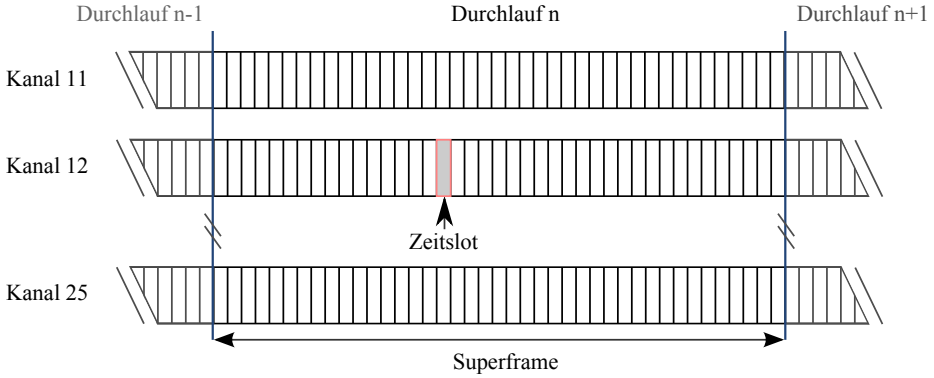


Abbildung 3.1.: TDMA kombiniert mit FDMA bei WirelessHART und ISA 100.11a

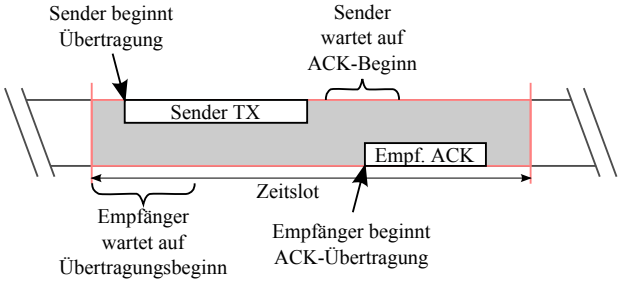


Abbildung 3.2.: Zeitlicher Verlauf eines Zeitslots bei WirelessHART und ISA 100.11a (nach [Eur10])

Der Network Manager (WirelessHART) bzw. System Manager (ISA 100.11a), ein zentraler Verwaltungsknoten, bestimmt, welche Knoten in einem bestimmten Zeitslot auf einem bestimmten Kanal senden und welche empfangen. Diese Zuordnung gilt für jede Wiederholung des Superframes, zu dem sie gehört. Daher bestimmt die Anzahl Zeitslots in einem Superframe das Intervall, indem eine Reservierung wiederkehrt. Auf einem Gerät können zudem mehrere Superframes konfiguriert sein, die sich periodisch wiederholen. Die Anzahl Zeitslots in einem Superframe sowie die Zuordnung der Zeitslots zu Knoten können sich je nach Superframe unterscheiden. ISA 100.11a erlaubt es zudem, dass ein Gerät zeitgleich an mehreren Superframes teilnimmt, wodurch ermöglicht werden soll, dass unterschiedliche Kommunikations-Schedules gleichzeitig bedient werden können. Da es dadurch dazu kommen kann, dass Zuordnungen mehrerer Superframes überlappen, werden derartige Konflikte über Prioritäten aufgelöst.

Ein Zeitslot wird im Normalfall genau einem Sender und einem Empfänger zugeordnet. Durch eine solche exklusive Reservierung werden Kollisionen verhindert und eine *deterministische Übertragung* sichergestellt[2].

Neben exklusiv reservierten Unicast-Übertragungen erlauben beide Standards auch Multicast-Übertragungen, bei denen einem Zeitslot ein Sender-Knoten und mehrere Empfänger-Knoten zugeordnet werden. In diesem Fall werden keine Acknowledgements gesendet, sodass der Sender nicht feststellen kann, ob die Nachricht fehlerfrei übertragen wurde.

ISA 100.11a erlaubt darüber hinaus sog. *Duocast-Übertragungen* mit einem Sender und zwei Empfängern, die nacheinander den fehlerfreien Empfang per Acknowledgement bestätigen. Der zeitliche Verlauf eines Duocast-Zeitslots ist in Abb. 3.3 dargestellt. Der erste Empfänger einer Duocast-Übertragung ist in der Zieladresse des Rahmens angegeben und weiß daher, dass er zuerst sein Acknowledgement senden muss.

[2]Vorausgesetzt, dass keine externen Störquellen das Medium belegen.

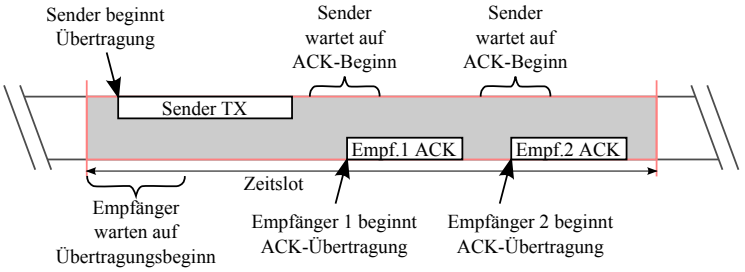


Abbildung 3.3.: Ein Zeitslot bei ISA 100.11a unter Verwendung von Duocast

Duocast-Übertragungen sind für den Fall gedacht, dass ein Knoten in Reichweite von zwei sog. Backbone-Routern ist, die das drahtlose Netz mit einem drahtgebundenen Netz verbinden. Damit dieser Knoten eine Nachricht an das Backbone weitergibt, ist es unerheblich, an welchen der beiden Backbone-Router er die Nachricht erfolgreich übermittelt. Mit Hilfe eines Duocasts kann er die Nachricht zeitgleich an beide Backbone-Router senden. Erhält er von mindestens einem Backbone-Router ein Acknowledgement, war die Übertragung erfolgreich. Somit wird die Wahrscheinlichkeit erhöht, dass die erste Übertragung zum Backbone erfolgreich verläuft, sodass weniger Neuübertragungen nötig werden. Falls der sendende Knoten das erste Acknowledgement erfolgreich erhalten hat, muss er das zweite nicht abwarten und kann so Energie sparen. Die Abstimmung der Backbone-Router untereinander zur Vermeidung einer doppelten Übertragung im Backbone wird nicht vom ISA 100.11a Standard geregelt.

Das Senden eines zweiten Acknowledgements benötigt zusätzliche Zeit, daher muss bei der Verwendung von Duocasts die Länge eines Zeitslots um typischerweise 1 bis 2 ms erhöht werden. Im Gegensatz zu WirelessHART, welches eine fixe Zeitslot-Länge von 10 ms festlegt, erlaubt ISA 100.11a es daher, die Zeitslot-Länge innerhalb eines Superframes zu konfigurieren. Obwohl explizit erlaubt wird, dass Superframes mit unterschiedlicher Zeitslot-Länge gleichzeitig in einem Netzwerk genutzt werden, gibt der Standard an, derartige Konfigurationen nicht näher betrachtet zu haben, und verlangt von Geräten die Unterstützung dafür nicht. Prinzipiell kann die Länge eines Zeitslot zwischen 0 und 62,5 ms in Schritten von 2^{-20} s (ca. 0,95 µs) konfiguriert werden. Welche Längen tatsächlich möglich sind, dürfte allerdings auch vom Gerät abhängig sein.

Neben exklusiv reservierten Slots erlauben beide Standards sog. *Shared Slots*, die mehreren Sendern zugeordnet werden. Hierbei sind Kollisionen nicht ausgeschlossen und daher keine deterministische Übertragung mehr sichergestellt. Ob eine Kollision aufgetreten ist, wird am Ausbleiben des Acknowledgements erkannt. Bei WirelessHART wird ein Backoff-Mechanismus eingesetzt, um die Wahrscheinlichkeit zu verringern, dass es bei Neuübertragungen erneut zu Kollisionen kommt. ISA 100.11a erlaubt es dagegen, unterschiedlichen Knoten unterschiedliche Prioritäten in einem Shared Slot zu geben, indem der Zeitpunkt, zu dem die Übertragung frühestens beginnen darf, konfiguriert werden kann. Da das Abhören des Mediums vor dem Senden zusätzliche Zeit benötigt, kann man bei ISA 100.11a die Slotdauer hierfür erhöhen.

Eine weitere Variante des wettbewerbsbasierten Zugriffs bietet ISA 100.11a mit dem *Slow Hopping* Mechanismus. Eine Slow Hopping Periode ist dabei eine Zeitspanne, die sich über mehrere Zeitslots erstreckt. Innerhalb der Slow Hopping Periode müssen Sender die Zeitslot-Grenzen nicht beachten und können prinzipiell jederzeit beginnen zu senden. Sie müssen lediglich die Belegung des Mediums zunächst per CCA prüfen. Die einer Slow Hopping Periode zugewiesenen Empfänger müssen während der gesamten Periode das Medium abhören, was zu einem erhöhten Energiebedarf führt. Als Motivation für diesen Mechanismus nennt der ISA 100.11a

Standard insbesondere, dass es so möglich wird, Knoten mit einer durch lange Schlafzeiten bedingten schlechteren Zeitsynchronisation das Senden zu ermöglichen. Da die Knoten sich innerhalb einer Slow Hopping Periode nicht genau an Zeitslot-Grenzen halten müssen, reicht eine ungenauere Uhrensynchronisation zum Senden aus, ohne dass die Gefahr besteht, in für andere Knoten exklusiv reservierten Zeitslots zu senden. Der Wechsel zwischen Slow Hopping und Slotted Hopping ist möglich und wird als *Hybrid Hopping* bezeichnet.

Da beide Standards slotted TDMA zum Medienzugriff einsetzen, ist eine genaue Tick- oder Zeitsynchronisation erforderlich. Dazu senden beide Standards mit sämtlichen regulären Rahmen (sowohl Daten- als auch Acknowledgement-Rahmen) einen Zeitstempel, anhand dessen eine Uhrenkorrektur vorgenommen wird. Es werden also nicht wie beispielsweise im von ZigBee übernommenen IEEE 802.15.4 MAC Layer besondere Beacon-Rahmen zur Zeitsynchronisation gesendet.

Wie bereits erwähnt, setzen sowohl WirelessHART als auch ISA 100.11a ein *Channel Hopping* ein, um die Zuverlässigkeit der Übertragung zu erhöhen. Dabei verschiebt sich mit jeder Wiederholung eines Superframes die Kanalzuordnung nach einem vom Network bzw. System Manager vorgegebenen Muster, wie es in Abb. 3.4 beispielhaft dargestellt wird. Falls ein Kanal häufiger gestört wird als andere, wird dadurch nicht immer wieder die gleiche Übertragung gestört. Es erhöht sich also die Wahrscheinlichkeit, dass eine Übertragung im nächsten Superframe erfolgreich ist. Falls bestimmte Kanäle generell unzuverlässig sind, beispielsweise weil ein WLAN-Netzwerk diese Frequenzen nutzt, so können diese durch *Blacklisting* aus dem Channel Hopping ausgeschlossen werden. Bei WirelessHART muss dazu der Netzwerkadministrator problematische Kanäle manuell konfigurieren. ISA 100.11a erlaubt es darüber hinaus, dass Knoten selbstständig problematische Kanäle erkennen und diese nicht für weitere Übertragungen nutzen. Das bedeutet aber lediglich, dass sie den Zeitslot auf diesem Kanal nicht nutzen, es wird nicht automatisch ein Zeitslot auf einem anderen Kanal bereitgestellt. So wird zwar verhindert, dass ISA 100.11a Geräte anderer Netzwerke wie beispielsweise WLAN dauerhaft stören, allerdings führt dies auch dazu, dass reservierte Slots nicht genutzt werden und Übertragungen nicht oder verzögert (auf einem anderen Kanal) stattfinden.

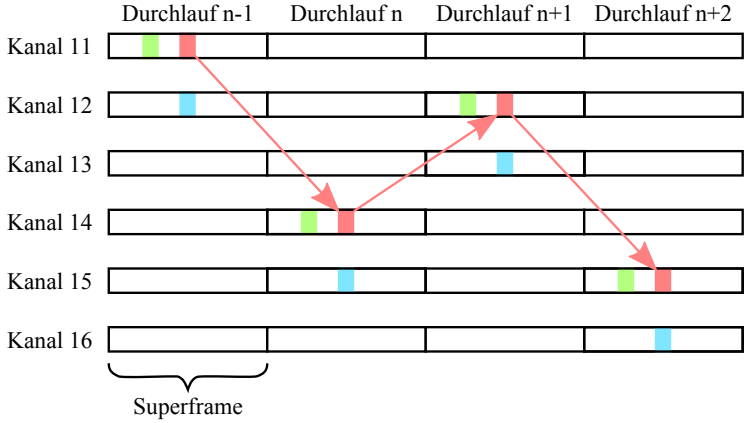


Abbildung 3.4.: Channel Hopping in WirelessHART und ISA 100.11a (nach [Eur10])

3.2.3 Network Layer

Als Aufgabe des Network Layers, auch *Vermittlungsschicht* genannt, wird die Vermittlung von Datenpaketen von einem Start- zu einem Zielknoten gesehen (vgl. OSI-Modell [Int94]). Dies

schließt insbesondere *Routing*, also das Finden eines geeigneten Pfades vom Start- zum Zielknoten, sowie ggf. *Fragmentierung* mit ein.

Die Aufgabe des Routings wird vom ISA 100.11a Standard aufgeteilt: Routing innerhalb eines ISA 100.11a-Funknetzwerks wird nicht wie vom OSI-Modell beschrieben vom Network Layer übernommen, sondern bereits von der darunter liegenden Schicht, dem sog. Data Link Layer, der auch die MAC-Schicht enthält. Der eigentliche Network Layer ist nur für das Routing über ein Backbone, beispielsweise in ein drahtgebundenes Netzwerk hinein, zuständig. Das Backbone und daran angebundene Netze werden allerdings im Rahmen des ISA 100.11a Standards nicht genauer definiert. Somit übernimmt der Network Layer in erster Linie die Übersetzung von kurzen 16 Bit Adressen in lange 128 Bit Adressen, die von der Anwendung und über das Backbone hinaus genutzt werden können, sowie die Fragmentierung von Datenpaketen, die nicht in einen Rahmen der darunter liegenden Schicht passen.

Obwohl ISA 100.11a das Routing innerhalb des Funknetzwerks nicht als Aufgabe des Network Layers spezifiziert, wird das im Data Link Layer spezifizierte Routing im Folgenden im Vergleich zum Routing von WirelessHART betrachtet, welches wie üblich im Network Layer spezifiziert ist. Bei WirelessHART ist der Network Layer allerdings nicht nur für das Routing innerhalb des Funknetzwerks zuständig, sondern muss auch das Routing über Gateways hinweg in ein drahtgebundenes HART-Netzwerk unterstützen.

Beide Standards beschreiben zwei unterschiedliche Routing-Verfahren: Das Source- und das Graph-Routing. Für beide Verfahren werden die Routen vom System- bzw. Network-Manager vorberechnet und an die Knoten verteilt, welche diese speichern und bei Bedarf einsetzen. Der Network- bzw. System-Manager bekommt bei beiden Standards von den Knoten regelmäßig Reports geschickt, in denen die Qualität der Links bewertet wird. Basierend auf diesen Informationen kann er bei Bedarf Routen neu berechnen und verteilen.

Beim *Source-Routing* wählt der Startknoten eine der gespeicherten Routen zum Zielknoten und vermerkt sie anhand ihrer ID im zu übertragenden Paket. Alle Knoten entlang der Route können anhand dieser ID die Route nachschlagen und so bestimmen, an welchen Knoten sie das Paket weiterleiten müssen. Ist die Übertragung auf einem der in der Route verwendeten Links nicht möglich, geht das Paket verloren.

Eine zuverlässigere Übertragung, die auch bei einzelnen gestörten Links funktioniert, bietet das *Graph-Routing*. Eine Graph-Route ist eine Teilmenge der vorhandenen Links des Netzwerks und wird ebenfalls anhand einer ID identifiziert. Genau wie Source-Routes werden Graph-Routes vom Network- bzw. System-Manager berechnet und an die Knoten im Netzwerk verteilt. Für ein Paket, dass per Graph-Routing verschickt wird, hat jeder Knoten, der das Paket weiterleitet, eine *Menge* möglicher Folgeknoten. Wird die Übertragung zu einem Folgeknoten gestört, kann er einen anderen Folgeknoten aus der Menge auswählen. Eine Graph-Route definiert also nicht nur einen Pfad vom Start- zum Zielknoten, sondern eine Reihe unterschiedlicher Pfade, die je nach Netzsituation dynamisch genutzt werden.

Fragmentierung wird von WirelessHART auf dem Network Layer nicht unterstützt. Allerdings bietet der Application Layer mit sogenannten *Block Transfers* Funktionalität zur Fragmentierung und Wiederzusammenfügung von großen Datenpaketen.

3.2.4 Transport Layer

Auf dem Transport Layer unterstützt WirelessHART anders als ISA 100.11a optional Acknowledgements zu Transaktionen, also die Bestätigung des Empfangs einer Übertragung über meh-

rere Hops hinweg. ISA 100.11a stellt dagegen nur einen verbindungslosen Dienst bereit, der auf dem User Datagram Protocol (UDP) [Pos80] über IPv6 [DH98] basiert. Darüber hinaus nutzt der Transport Layer von ISA 100.11a Funktionen zur Ende-zu-Ende-Verschlüsselung, zur Integritätsprüfung und zur Kompression aus dem 6LoWPAN Standard [HT11]. In WirelessHART wird die Aufgabe der Ende-zu-Ende-Verschlüsselung zwar vom Network Layer übernommen, um einen besseren Vergleich mit ISA 100.11a zu ermöglichen, aber hier besprochen.

Beide Standards verwenden zur Verschlüsselung AES-CCM-128. Dabei kommen unterschiedliche symmetrische Schlüssel zum Einsatz, die vom Network- bzw. System-Manager an die Knoten verteilt werden. Der sichere Beitritt eines neuen Knotens zum Netz wird gewährleistet, indem neue Knoten über einen speziellen *Join Key* verfügen müssen. ISA 100.11a bietet allerdings optional die Möglichkeit, dass sich neue Knoten über einen *Global Key* anmelden können, einem allgemein bekannten Schlüssel, der keine Sicherheit bietet. So am Netz angemeldete Knoten können keine Ende-zu-Ende-Verschlüsselung nutzen. Diese Option ermöglicht es Herstellern, einfachere Knoten mit geringerem Sicherheitsstandard zu entwickeln. Die Ende-zu-Ende-Verschlüsselung der Kommunikation zwischen zwei Knoten wird mit *Session Keys*, die für jede paarweise Kommunikation unterschiedlich sind, durchgeführt. Neben symmetrischen Schlüsseln unterstützt ISA 100.11a (anders als WirelessHART) optional auch asymmetrische Verschlüsselung.

3.2.5 Application Layer

Der Application Layer von WirelessHART übernimmt prinzipiell den des drahtgebundenen HART-Protokolls und ist damit zu HART kompatibel. Dies vereinfacht die Verknüpfung von drahtgebundenen HART-Geräten mit drahtlosen WirelessHART-Geräten. Sämtliche Kommunikation erfolgt über eine Reihe von definierten Kommandos, die sich in folgende Klassen aufteilen lassen [CNM10]:

- *Universal Commands* sind Basis-Kommandos, die von allen HART-kompatiblen Geräten unterstützt werden müssen.
- *Common Practice Commands* sind Kommandos, die von einer breiten Palette von HART-Geräten sinnvoll einsetzbar sind und daher von HART-Geräten soweit möglich unterstützt werden sollten.
- *Non-public Commands* sollten nur während der Entwicklung eines HART-Geräts und nicht im produktiven Einsatz verwendet werden.
- *Wireless Commands* sind Kommandos für Drahtlosknoten, die von allen WirelessHART-Geräten unterstützt werden müssen.
- *Device Family Commands* sind Sammlungen von Kommandos, welche die Installation und Konfiguration von Feldgeräten unterstützen, ohne dass die Unterstützung gerätespezifischer Kommandos nötig ist.
- *Device-Specific Commands* sind für ein Gerät spezifische Kommandos, die vom Hersteller definiert werden.

ISA 100.11a setzt dagegen nicht auf ein kommandobasiertes Protokoll, sondern auf ein objektorientiertes. Der Ansatz, Objekte der realen Welt als Software-Objekte zu modellieren, ist aus der objektorientierten Programmierung bekannt und hat sich dort etabliert. Die Wahl des objektorientierten Modells für die Anwendungsschicht in ISA 100.11a begründet der Standard damit,

dass das Modell „allgemein anerkannte architektonische Prinzipien der logischen Informationstrennung unterstützt“[3]. Da alle Operationen auf Objekten ausgeführt werden, muss in jeder Nachricht zusätzlich die Kennung des zugehörigen Objekts enthalten sein. Der dadurch u.U. entstehende Overhead von mindestens einem Byte für die Objekt-Kennung[4] wird vom Standard als gerechtfertigt angesehen, um die architektonische Informationstrennung zu ermöglichen. Weiter wird argumentiert, dass eine kommandobasierte Anwendung auf das objektorientierte Modell abgebildet werden kann, indem die Kommandos als Objektmethoden beschrieben werden.
Wie in der objektorientierten Programmierung können mehrere Objekte Instanzen der selben Klasse sein. Auf jedem Knoten des Netzwerks können mehrere Applikationsprozesse laufen, welche mehrere Objekte enthalten können. Ein Objekt ist dabei stets genau einem Knoten zugeordnet. Um eine Methode auf einem entfernten Objekt auszuführen, wird eine Nachricht an den Knoten gesendet, der das Objekt enthält. Neben Methoden besitzen Objekte auch Attribute, welche von anderen Knoten gelesen und evtl. auch geschrieben werden können. Jedes Attribut kann einer der folgenden fünf Klassen zugeordnet werden:

- *Constant*: Diese Attribute dürfen ihren Wert während des Betriebs nicht verändern. Als Beispiel wird die Seriennummer eines Drahtlosknotens genannt. Der Wert von konstanten Attributen soll erhalten bleiben, wenn ein Gerät neu gestartet, vom Strom getrennt oder zurückgesetzt wird.
- *Static*: Ein statisches Attribut ändert seinen Wert selten. Eine Änderung wird normalerweise durch einen externen Auslöser hervorgerufen, beispielsweise durch ein Konfigurationswerkzeug. Als Beispiele werden u.a. Betriebsbereiche und Einheiten genannt. Die Werte statischer Attribute sollen erhalten bleiben, wenn ein Gerät neu gestartet oder vom Strom getrennt wird.
- *Static-volatile*: Dies sind statische Attribute, deren Wert nicht erhalten bleiben muss, wenn ein Gerät neu gestartet oder vom Strom getrennt wird.
- *Dynamic*: Ein dynamisches Attribut kann durch das Gerät, welches das Objekt enthält, spontan geändert werden. Als Beispiele werden Prozessvariablen, Ergebnisse von Berechnungen sowie Timer genannt. Werte dynamischer Objekte müssen nicht erhalten bleiben, wenn ein Knoten neu gestartet oder vom Strom getrennt wird.
- *Non-bufferable*: Diese Attribute dürfen nicht zwischengespeichert werden, d.h. wenn der Wert eines solchen Attributs abgefragt wird, muss er von dem Knoten abgefragt werden, der das Objekt enthält. Es ist also nicht erlaubt, dass ein Zwischenknoten die Anfrage aus dem Cache beantwortet. Diese Art von Attributen ist für kritische Sicherheitsinformation oder Werte gedacht, die sich so häufig ändern, dass Cachen nicht sinnvoll ist.

Neben nativen ISA 100.11a Applikationen bietet der Standard Unterstützung für das Tunneling anderer Protokolle. Dazu existiert ein spezieller Prozess, der einen oder mehrere Tunneling-Objekte enthalten kann und Unterstützung zur Protokoll-Übersetzung bereitstellt. Darüber hinaus schlägt der Standard vor, dass andere Protokoll-Konsortien wie die HART Communication Foundation ein Mapping von ihrem Protokoll auf das native ISA 100.11a Anwendungsprotokoll definieren, sodass eine Anwendung beispielsweise zunächst im HART-Protokoll entworfen und dann auf das ISA 100.11a-Protokoll abgebildet wird.

[3] Eigene Übersetzung aus [ISA11]. Originalwortlaut: „The object model supports well accepted architectural principles of logical information separation.“

[4] Objekt-Kennungen können mit mindestens 4 und maximal 16 Bit kodiert werden, da stets das Quell- und das Zielobjekt angegeben wird, ergibt sich so ein Mindest-Overhead von einem Byte.

3.2.6 Fazit

Die beiden Standards haben sehr viele Gemeinsamkeiten, unterscheiden sich allerdings im Detail teilweise deutlich. Die Gemeinsamkeiten beginnen bei dem Physical Layer, den beide auf IEEE 802.15.4 aufbauen. Daraus ergibt sich, dass beide Standards dasselbe 2,4 GHz Band nutzen, die selbe Bruttodatenrate von 250 kbit/s sowie die selbe Reichweite von ca. 30 m im Innenraum und 70 m im Außenbereich aufweisen. Auf MAC-Ebene kombinieren beide Standards TDMA mit FDMA und nutzen Channel Hopping sowie Channel Blacklisting zur Zuverlässigkeits-Steigerung. Beide Standards setzen auf Mesh-Topologien und nutzen das Source- bzw. Graph-Routing-Verfahren.

Bei den Unterschieden zeigt sich generell, dass WirelessHART mehr eindeutige Vorgaben macht, wohingegen ISA 100.11a mehr Flexibilität zulässt. Auf MAC-Ebene erlaubt ISA 100.11a beispielsweise eine konfigurierbare Slotlänge, wohingegen WirelessHART diese mit 10 ms fest vorgibt. Außerdem bietet ISA 100.11a mit Duocast-Übertragungen und Slow-Hopping Funktionen, die WirelessHART nicht bietet. Während WirelessHART keine Unterscheidung zwischen eingeschränkten Knoten und Knoten mit vollem Funktionsumfang erlaubt, können ISA 100.11a Geräte beispielsweise auf volle Routing-Funktionalität verzichten oder keine Unterstützung für Ende-zu-Ende-Verschlüsselung bieten. Damit bietet ISA 100.11a die Möglichkeit, einfachere und damit billigere bzw. batteriebetriebene Knoten zu entwickeln. Auf dem Transport Layer versucht ISA 100.11a mit UDP, IPv6 und 6LoWPAN andere offene Standards zu integrieren, um eine Interoperabilität mit anderen Systemen zu erleichtern. WirelessHART dagegen zielt auf eine möglichst gute Kompatibilität zu drahtgebundenen HART-Geräten ab. Dies zeigt sich vor allem auf dem Application Layer, der bei WirelessHART auf dem drahtgebundenen HART-Protokoll basiert, wohingegen ISA 100.11a ein möglichst flexibles objektorientiertes Protokoll definiert und dabei auch die Möglichkeit vorsieht, andere Protokolle wie das HART-Protokoll zu tunneln.

Prinzipiell ist die größere Flexibilität von ISA 100.11a aus Anwendersicht sicher wünschenswert. Allerdings führt die höhere Flexibilität bei der Installation des Drahtlosnetzwerks auch zu einem höheren Konfigurationsaufwand, wohingegen WirelessHART-Systeme eher nach dem Plug & Play Prinzip schnell und einfach einsetzbar sind. Darüber hinaus könnte die hohe Flexibilität dazu führen, dass Geräte-Hersteller zunächst nur einen eingeschränkten Umfang implementieren. Dies könnte zum einen bedeuten, dass die Vorteile von ISA 100.11a nicht nutzbar werden, zum anderen, dass unterschiedliche Hersteller einen unterschiedlichen Umfang des Standards umsetzen und daher Hardware nicht voll kompatibel ist.

Da nicht absehbar ist, dass einer der beiden Standards sich durchsetzt und den anderen verdrängt, müssen Anwender abwägen, ob sie die höhere Flexibilität von ISA 100.11a bevorzugen bzw. benötigen oder die einfachere Inbetriebnahme von WirelessHART. In der Praxis wird die Entscheidung oft dadurch beeinflusst sein, welche Hardware im gegebenen Anwendungsbereich verfügbar ist. Da das Angebot von WirelessHART-kompatiblen Geräten immer noch größer zu sein scheint als das von ISA 100.11a-Geräten, wird dies in vielen Fällen für den Einsatz von WirelessHART sprechen.

4. KAPITEL

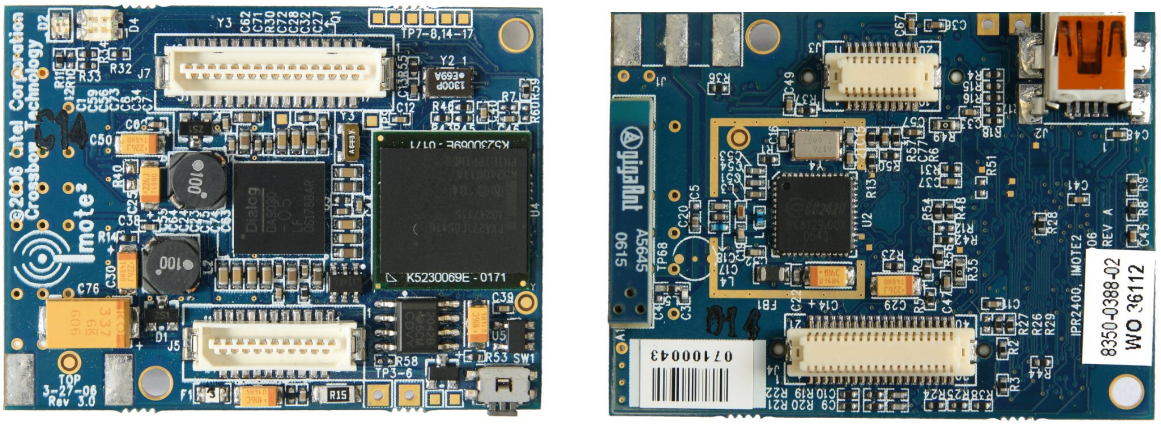
Konzeption eines drahtlosen Kommunikationssystems für den Produktionsbereich

In Kapitel 3 wurden mit WirelessHART und ISA 100.11a zwei Standards vorgestellt, die für den Einsatz als drahtloses Kommunikationssystem im Produktionsbereich entwickelt wurden. In diesem Kapitel wird in Hinblick auf die in Kapitel 2 formulierten Anforderungen ein Konzept für ein neues drahtloses Kommunikationssystem entworfen.

Das Konzept wird nicht nur in der Theorie entworfen, sondern auch, teilweise im Rahmen dieser Arbeit (s. Kapitel 5), teilweise im Rahmen anderer Arbeiten, für reale Hardware implementiert. Daher wird zunächst kurz die eingesetzte Hardware beschrieben. Anschließend werden die unterschiedlichen Lösungsmöglichkeiten auf den einzelnen Schichten des Systems betrachtet, ihre Vor- und Nachteile untersucht und jene Alternativen ausgewählt, die am besten zur Erfüllung der gegebenen Anforderungen beitragen. Abschließend werden die getroffenen Entscheidungen zusammengefasst.

4.1 Eingesetzte Hardware

Zum Einsatz kommt die von Intel entwickelte Imote 2-Plattform [Cro09]. Abbildung 4.1 zeigt ein Foto von Ober- und Unterseite dieses Sensorknotens. Auf der Oberseite ist der von Intel entwickelte PXA271-Prozessor mit ARM-kompatibler XScale-Architektur zu sehen. Auf der Unterseite ist der von Chipcon entwickelte CC2420-Transceiver [Chi07] verbaut, über den der Knoten drahtlos mit anderen kommunizieren kann. Dieser Transceiver ist für die Übertragung von Rahmen nach dem IEEE 802.15.4-Standard auf dem 2,4 GHz Band ausgelegt. Der Physical Layer sowie Teile des MAC Layers sind direkt in Hardware integriert. Auf den Knoten können Sensorboards aufgesteckt werden, welche Sensoren, beispielsweise für Temperatur oder Licht, besitzen. Außerdem ist es möglich, über GPIO (General Purpose Input Output), SPI (Serial Peripheral Protocol) oder I²C (Inter-Integrated Circuit) weitere Sensoren mit dem Board zu verbinden. Die Stromversorgung des Knotens kann über USB oder über aufgesteckte Batterie- bzw. Akku-Boards erfolgen. Die Abmessungen der Platine betragen ohne aufgesteckte Sensor- oder Batterie-Boards 48 mm x 36 mm x 9 mm, das Gewicht 12 g. Damit erfüllt die Hardware die in Kapitel 2.8 formulierten Anforderungen an Größe und Gewicht.



(a) Oberseite mit PXA271-Prozessor (rechts)

(b) Unterseite mit CC2420-Transceiver (Mitte) und Antenne (links)

Abbildung 4.1.: Imote 2-Sensorknoten

4.2 Protokoll-Architektur

Wie bei Kommunikationsprotokollen üblich, wird das hier entwickelte Protokoll in einer Schichtenarchitektur aufgebaut. Die einzelnen Schichten sind in Abb. 4.2 veranschaulicht. Die unteren drei Schichten (Physical Layer, MAC Layer und Network Layer) entsprechen ähnlich wie bei WirelessHART und ISA 100.11a in etwa den unteren drei Schichten des OSI Referenz-Modells für Netzwerkprotokolle [Int94]. Die darüber liegende Schicht wird als *Middleware Layer* bezeichnet. Sie stellt die Schnittstelle zur Anwendung dar und bietet der Anwendung einen dienstorientieren Kommunikationsansatz, wie er in den Anforderungen gefordert wurde (s. Kapitel 2.6). Sie erfüllt damit unter anderem Funktionalitäten, die im OSI-Modell der Transport- und Sitzungsschicht zugeschrieben werden. Die Anwendung selbst ist nicht Teil des Protokolls und wird daher hier nicht genauer betrachtet.

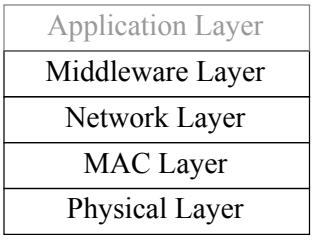


Abbildung 4.2.: Schichtenarchitektur des konzipierten Protokolls

Im Folgenden werden für die unterschiedlichen Schichten die möglichen Alternativen beschrieben und bewertet. Ausgewählt wird schließlich die Lösungsmöglichkeit, die am ehesten dazu beiträgt, die in Kapitel 2 formulierten Anforderungen zu erfüllen. Dort, wo eine andere Lösung gewählt wird als bei WirelessHART bzw. ISA 100.11a, werden die Unterschiede betrachtet.

4.2.1 Physical Layer

Obwohl der CC2420-Transceiver den Einsatz des IEEE 802.15.4 Physical Layers bereits vorgibt, betrachten wir kurz auch mögliche Alternativen und deren Vor- und Nachteile.

Der Physical Layer von IEEE 802.15.4 wurde bereits in Kapitel 3.1.3 näher beschrieben. Er ist vor allem deshalb für drahtlose Kommunikationssysteme im Produktionsbereich geeignet, weil er einen sinnvollen Batteriebetrieb ermöglicht. Nur so können alle Vorteile eines drahtlosen Kommunikationssystems gegenüber einem drahtgebundenen ausgenutzt werden. Andere Standards wie IEEE 802.11 (WLAN) oder IEEE 802.15.1 (Bluetooth) bieten zwar deutlich höhere Datenraten, scheiden aber dadurch aus, dass ihr Energieverbrauch höher ist und dadurch kein Batteriebetrieb über einen längeren Zeitraum möglich ist. Mit 250 kbit/s im 2,4 GHz-Band bietet IEEE 802.15.4 eine für diese Anwendung ausreichende Brutto-Übertragungsrate (s. Kapitel 2.2). Die erzielte Reichweite von üblicherweise 30 m ist ebenfalls ausreichend.

Da im Produktionsumfeld eine zuverlässige Übertragung wichtig ist, bietet die Nutzung des 2,4 GHz Bandes allerdings auch Nachteile. In den letzten Jahren stieg der Absatz von WLAN-Geräten wie Smartphones [Staa] und Tablets [Stab] sowie die Verbreitung von öffentlichen Hotspots [Stac]. Mit der zunehmenden Verbreitung von WLAN wird das 2,4 GHz Band immer stärker genutzt und Interferenzen damit immer wahrscheinlicher. Der Einsatz von UWB verspricht dagegen im Vergleich eine zuverlässigere Übertragung und ist mit IEEE 802.15.4a bereits seit 2007 Teil des IEEE 802.15.4-Standards (s. Kapitel 3.1.5). Allerdings ist derzeit noch keine mit dem Imote 2 vergleichbare Hardwareplattform mit UWB-Unterstützung verfügbar, sodass der Einsatz von UWB hier nicht möglich ist.

4.2.2 MAC Layer

4.2.2.1 Multiple Access Verfahren

Die wichtigste Aufgabe des MAC Layers ist die Regelung des Zugriffs mehrerer Knoten auf das Funkmedium (*Multiple Access*). Damit es, wenn mehrere Knoten senden möchten, nicht zu Kollisionen kommt, muss der Zugriff auf das Medium kontrolliert ablaufen, wozu es unterschiedliche Verfahren gibt [SS12]:

- *Time Division Multiple Access (TDMA)*: Der Zugriff auf das Medium erfolgt zeitlich versetzt. Es belegt also zu einer Zeit immer nur ein Knoten das Medium. In der Regel wird die Zeit in Zeitslots aufgeteilt und ein Zeitslot genau einem Knoten zugewiesen, der damit in diesem Zeitslot die Möglichkeit zum Senden erhält.
- *Space Division Multiple Access (SDMA)*: Der Zugriff auf das Medium erfolgt räumlich getrennt. Es können also zwei Knoten das Medium gleichzeitig belegen, wenn sie so weit voneinander entfernt sind, dass keine Interferenzen auftreten können. Der Raum kann beispielsweise in Zellen aufgeteilt werden und es darf immer nur ein Knoten in einer Zelle senden.
- *Frequency Division Multiple Access (FDMA)*: Der Zugriff auf das Medium erfolgt auf unterschiedlichen Frequenzen. Dazu wird das Medium in Kanäle aufgeteilt und unterschiedlichen Knoten unterschiedliche Kanäle zugeordnet.
- *Code Division Multiple Access (CDMA)*: Der Zugriff auf das Medium erfolgt mithilfe unterschiedlicher Codes. Dabei nutzen mehrere Knoten gleichzeitig dasselbe Frequenzband und verwenden unterschiedliche Spreizcodes, anhand derer ihre Signale vom Empfänger wieder unterschieden werden können.

Oft werden die Verfahren kombiniert. So verwenden WirelessHART und ISA 100.11a beispielsweise eine Kombination aus TDMA, FDMA und SDMA: Prinzipiell erfolgt der Zugriff auf das

Medium über reservierte Zeitslots, allerdings werden zur selben Zeit Zeitslots auf unterschiedlichen Kanälen unterschiedlichen Knoten zugeordnet (s. Abb. 3.1). Außerdem können Knoten, die weit genug voneinander entfernt sind, im selben Zeitslot auf dem selben Kanal senden. Eine derartige Kombination erlaubt es, das Medium effizienter auszunutzen und so die Übertragungsrate des Gesamtnetzes zu verbessern.

Damit Kollisionen ausgeschlossen sind, muss die Zuweisung von Zeitslots und Kanälen exklusiv erfolgen. Dazu kann entweder eine zentrale Instanz oder ein dezentrales Zuteilungsprotokoll wie das Bit-Map-Protokoll [Kow02] verwendet werden. Die Einhaltung der geforderten Dienstgüte kann allerdings leichter garantiert werden, wenn die Vergabe von Reservierungen zentral kontrolliert wird.

Zur Kollisionsverhinderung kann neben exklusiven Reservierungen auch *Carrier Sense Multiple Access/Collision Avoidance* (CSMA/CA) verwendet werden. Dabei hört ein Knoten das Medium ab, bevor er zu senden beginnt und belegt es nur, wenn er es als frei erkennt. Dies hat den Vorteil, dass keine exklusiven Reservierungen nötig sind, und kann darüber hinaus zu einer effizienteren Mediennutzung führen. Dies ist insbesondere dann der Fall, wenn statt periodischem Verkehr viel sporadischer Verkehr auf dem Medium stattfindet. Allerdings kann dieses Verfahren Kollisionen nicht verhindern und ist daher nicht deterministisch. Hat ein Knoten das Medium als frei erkannt, muss er vom Empfangs- in den Sendemodus wechseln und kann währenddessen das Medium nicht abhören. Beginnt in dieser Zeit ein anderer Knoten mit dem Senden oder beginnen zwei Knoten exakt gleichzeitig, kommt es zur Kollision. Die statistische Wahrscheinlichkeit von Kollisionen kann zwar durch Verfahren wie CSMA/CA verringert, Kollisionen aber nicht komplett ausgeschlossen werden.

Für den Einsatz als Kommunikationssystem im Produktionsbereich ist (reines) CSMA/CA daher nicht geeignet. Wie in den Anforderungen in Kapitel 2.4 beschrieben wurde, muss das Kommunikationssystem eine zuverlässige Übertragung garantieren können, was durch CSMA/CA nicht gewährleistet werden kann. Darüber hinaus führt der Einsatz von CSMA/CA meist zu einem höheren Energieverbrauch. Da der Empfänger nicht weiß, wann mit dem Empfang eines Rahmens zu rechnen ist, muss er das Medium permanent abhören. Da Transceiver aber auch im Empfangsmodus einen bedeutenden Energieverbrauch haben, wird viel Energie verschwendet, um ein unbelegtes Medium abzuhören (Idle Listening [YHE02]) oder Rahmen zu empfangen, die für andere Knoten bestimmt sind (Overhearing [YHE02]). Beim Einsatz von TDMA kann dagegen wie beispielsweise bei WirelessHART oder ISA 100.11a einem Zeitslot außer einem Sender auch der Empfänger zugeordnet werden. Dadurch, dass der Knoten nur noch während Zeitslots im Empfangsmodus sein muss, in denen er als Empfänger zugeordnet wurde, kann er in der restlichen Zeit im energiesparenden idle-Modus sein. Im Vergleich zu CSMA/CA ist also ein besseres *Duty Cycling* möglich. Auch der Sender spart Energie, da er das Medium vor dem Senden nicht abhören muss und Kollisionen ausgeschlossen und Neuübertragungen damit seltener sind.

Für den Einsatz von TDMA spricht außerdem, dass beim Einsatz im Produktionsbereich der Datenverkehr hauptsächlich aus der Übermittlung von Sensorwerten und Regelungsbefehlen besteht (s. Kapitel 2.1). Da es sich hierbei meist um Daten handelt, die periodisch in einem festen Intervall gesendet werden, eignen sich exklusiv reservierte Zeitslots bestens für diese Datenströme.

Die Verwendung von CDMA ist im betrachteten Einsatzbereich ebenfalls denkbar, allerdings mit IEEE 802.15.4 nicht umsetzbar. Aus diesem und den oben genannten Gründen wird für den

MAC Layer das TDMA-Verfahren in Kombination mit FDMA und SDMA zur Steigerung der Gesamt-Übertragungsrate des Netzwerks gewählt.

4.2.2.2 Slotting

Beim Einsatz von TDMA muss die Zeitaufteilung klar geregelt sein, damit alle Knoten wissen, welcher Zeitslot gerade aktiv ist und ob sie gerade senden dürfen oder nicht. Dazu erfolgt eine *Ticksynchronisation*, sodass alle Knoten einen gemeinsamen Takt haben. Anders als bei WirelessHART und ISA 100.11a, welche die Synchronisation an normalen Nutzrahmen vornehmen, wird hierzu *Black-Burst-Synchronization (BBS)* [GK08] eingesetzt. Dieses Verfahren sendet in regelmäßigen Abständen Tick-Rahmen, welche durch eine spezielle *Black-Burst-Kodierung* kollisionsgeschützt übertragen werden. Dadurch erzielt das Verfahren einen geringen und deterministisch begrenzten Tick-Offset. Diese deterministische Begrenzung ist notwendig, um garantieren zu können, dass bei exklusiven Reservierungen Kollisionen ausgeschlossen sind. Im Gegensatz zur Synchronisation an Nutzrahmen wie bei WirelessHART und ISA 100.11a ist BBS unabhängig von den höheren Schichten: Da die Synchronisation anhand spezieller Tick-Rahmen erfolgt, ist sie nicht darauf angewiesen, dass die Anwendung oft genug Nutzrahmen versendet, an denen eine Synchronisation durchgeführt werden kann. Der Einsatz des Verfahrens bietet sich darüber hinaus an, da eine optimierte Implementierung für die Imote 2-Plattform vorhanden ist [Eng13, ECG14].

Durch BBS wird ein regelmäßiger Tick vorgegeben. Zwischen zwei Ticks steigt auf Grund der unterschiedlichen Uhrenrate (Clock Skew) der Knoten die Uhrenabweichung (Clock Offset) an. Um den Overhead durch die Synchronisation gering zu halten, muss die Zeit zwischen zwei Ticks – d.h. das Resynchronisationsintervall – so groß gewählt werden, dass der entstehende maximale Offset im akzeptablen Bereich bleibt. In der Regel wählt man ein Resynchronisationsintervall von einigen wenigen Sekunden. Der Tick wird als *Makro-Tick* verwendet und die Zeit dazwischen in kleinere Zeitslots, sogenannte *Mikro-Slots*, aufgeteilt.

Damit sowohl längere als auch kürzere Reservierungen effizient möglich werden, wird die Länge eines Mikro-Slots sehr klein gewählt (z.B. 1 ms) und man erlaubt, dass eine Übertragung sich über mehrere Mikro-Slots erstrecken kann. Je nach Länge des zu übertragenden Rahmens werden mehr oder weniger Mikro-Slots reserviert (s. Abb. 4.3). In einem reservierten Bereich ist es möglich, dass derselbe Sender mehrere Rahmen sendet. Dieser Ansatz unterscheidet sich stark vom Slotting von WirelessHART und ISA 100.11a, bei dem stets maximal eine Übertragung (inkl. ACK) in einem Zeitslot stattfindet. Obwohl ISA 100.11a die Konfiguration der Slot-Länge erlaubt, ist dies sehr unflexibel, da die konfigurierte Slotlänge stets für den ganzen Superframe gilt. Mit dem hier gewählten Ansatz werden dagegen sehr flexibel sowohl lange als auch kurze Reservierungen in beliebiger Abfolge möglich.
Würde als Mikro-Slot-Dauer beispielsweise 1 ms gewählt, so hätte in dem in Abb. 4.3 dargestellten Bereich Knoten 1 eine Reservierung von 10 ms, Knoten 2 eine Reservierung von 5 ms, Knoten 3 eine Reservierung von 15 ms und Knoten 4 eine Reservierung von 6 ms Dauer.

Indem das Medium nur solange reserviert wird, wie zur Übertragung eines Rahmens benötigt, verringert sich der Verschnitt, der normalerweise entsteht, wenn die Übertragung des Rahmens vor dem Ende des Slots abgeschlossen ist. Dieser Ansatz erlaubt somit eine effizientere Nutzung des Mediums.

Die Länge eines Makro-Slots, also der Zeit zwischen zwei Makro-Ticks, kann beim Einsatz von BBS nicht beliebig erhöht werden, da die Synchronisationsungenauigkeit mit der Zeit zwischen

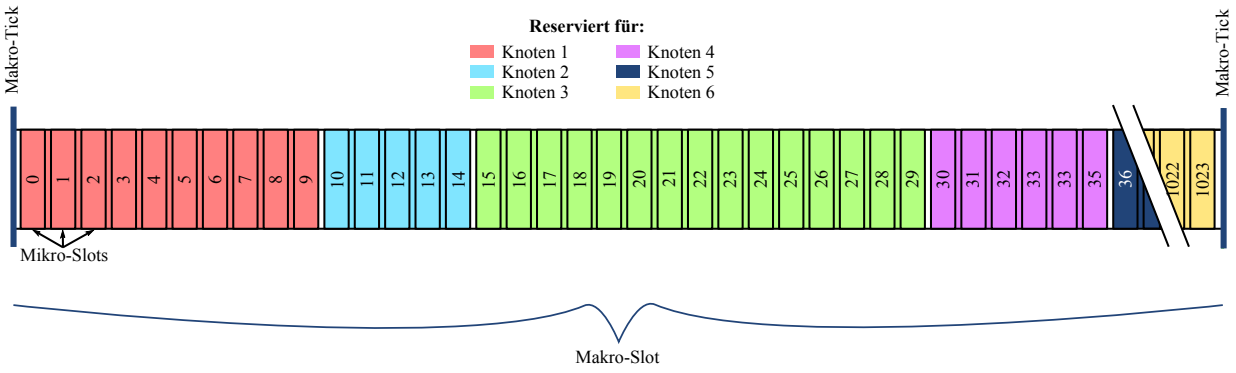


Abbildung 4.3.: Flexible Reservierungen mehrerer Mikro-Slots innerhalb eines Makro-Slots

zwei BBS-Ticks anwächst. Beträgt die Dauer eines Makro-Slots beispielsweise 2 s, so müsste man für eine Übertragung, die nur alle 6 s stattfindet, trotzdem in jedem Makro-Slot alle 2 s ausreichend Mikro-Slots reservieren. Zwei Drittel der reservierten Slots würden aber nicht genutzt. Um eine derart ineffiziente Nutzung des Mediums zu vermeiden, wird das Slotting um eine dritte Ebene erweitert: Mehrere Makro-Slots bilden einen *Super-Slot*. Ein Super-Slot wird von zwei Super-Ticks begrenzt, wobei es sich um Makro-Ticks handelt, die um ein Super-Tick-Flag erweitert wurden. Die Mikro-Slot-Zählung läuft über Makro-Slot-Grenzen hinaus innerhalb eines Super-Slots weiter (s. Abb. 4.4). Reservierungen kehren damit nicht mit jedem Makro-Slot, sondern mit jedem Super-Slot wieder. Wählt man wie in Abb. 4.4 eine Super-Slot-Länge von 3 Makro-Slots und eine Makro-Slot-Länge von 2 s, würde jede Reservierung alle 6 s wiederkehren. Indem die Länge des Super-Slots passend zu den Intervallen der auftretenden Datenströme gewählt wird, kann das Medium am effizientesten genutzt werden.

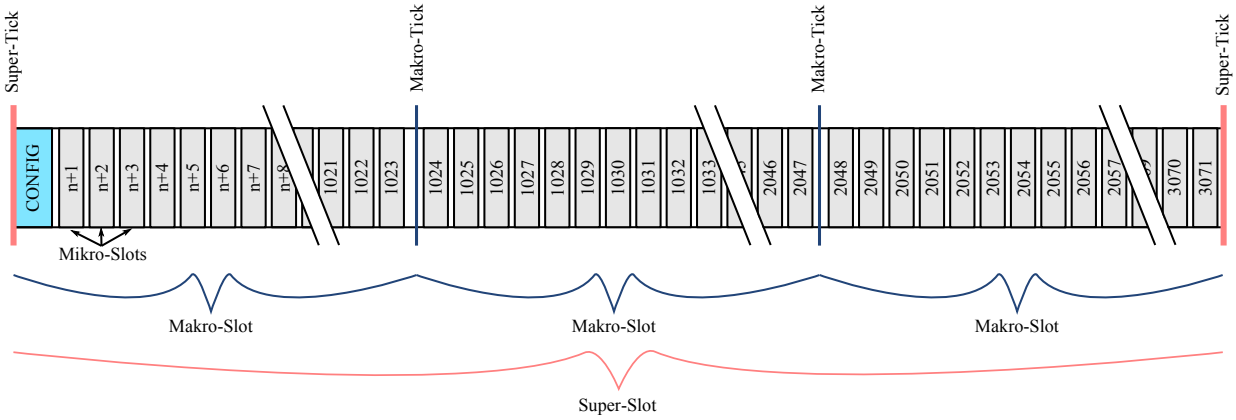


Abbildung 4.4.: Dreistufiges Slotting: Mikro-, Makro- und Super-Slots

Am Anfang eines Super-Slots ist außerdem vorgesehen, in einem CONFIG-Rahmen spezielle Konfigurationsdaten netzweit zu verbreiten (s. Abb. 4.4). Beispielsweise wird den Knoten auf diese Weise mitgeteilt, ob sich das System in der Startphase befindet, in der es sich selbst einmisst, oder bereits im Normalbetrieb die eigentliche Anwendung ausführt. Details dazu werden in Kapitel 4.2.3.2 erläutert.

4.2.2.3 Exklusive Reservierungen

Die exklusiven Reservierungen der Zeitslots sind nicht statisch konfiguriert, sondern können dynamisch angefordert werden. Dies ist nötig, damit die in den Anforderungen verlangte dynamische Anforderung von Diensten (s. Kapitel 2.6) auf höheren Ebenen unterstützt werden kann, ohne dass stark überreserviert werden muss. Im Folgenden wird kurz skizziert, wie dynamische Reservierungen umgesetzt werden können, allerdings wird dieser Aspekt hier nicht im Detail betrachtet.

Die Verwaltung der Reservierungen übernimmt ein zentraler Knoten, der *Network-Manager* genannt wird. Nachdem beim Start des Netzwerks dessen Topologie bestimmt wurde (s. Kapitel 4.2.3.2), sind zunächst alle Zeitslots exklusiv für den Network-Manager reserviert, alle anderen Knoten sind im Empfangsmodus. Der Network Manager weist nun jedem Knoten eine exklusive Reservierung zur Broadcast-Übertragung an alle unmittelbaren Nachbarn zu[5]. Über diese Reservierungen ist sichergestellt, dass alle Knoten ausreichend kommunizieren können, um weitere Reservierungen anzufordern. Im Folgenden können Knoten dynamisch weitere Reservierungen anfordern, indem sie eine entsprechende Anfrage (evtl. über andere Knoten) an den Network-Manager schicken. Dieser bestätigt die empfangenen Anfragen oder lehnt sie ab. Außerdem teilt er in regelmäßigen Abständen jedem Knoten mit, in welchen für ihn reservierten Zeitslots er senden darf und in welchen er empfangen muss. Die zu Beginn ausgegebenen Broadcast-Reservierungen werden von den Knoten wieder abbestellt, sobald sie genügend Reservierungen angefordert haben, um den weiteren Betrieb zu gewährleisten.

Entsprechend der Anforderungen müssen auch mobile Knoten unterstützt werden (s. Kapitel 2.6). Wenn für mobile Knoten Slots reserviert werden, ist zu beachten, dass durch die Bewegung der Knoten die Möglichkeiten für SDMA einschränkt sind. Im einfachsten Fall werden Zeitslots (auf einem Kanal) für mobile Knoten netzweit reserviert (kein SDMA). Hier besteht Raum für Optimierung, welche im Rahmen einer anderen Arbeit untersucht werden kann. Beispielsweise ist der Bewegungsbereich eines Roboters in einer Produktionsanlage i.d.R. stark eingeschränkt. Konfiguriert man diesen Bewegungsbereich, kann der Network-Manager bei der Anforderung einer exklusiven Reservierung für den mobilen Knoten diese Information nutzen, um SDMA mit Einschränkungen zuzulassen.

4.2.2.4 Mode-Based Scheduling with Fast Mode-Signaling

Der Einsatz exklusiver periodischer Reservierungen eignet sich besonders gut für Datenströme, die in einem gleichbleibenden Intervall Rahmen senden, wie beispielsweise einen Temperatur-Sensor (vgl. Anwendungsszenario in Kapitel 2.1). Ereignisgesteuerte Datenströme dagegen, wie beispielsweise der Steuerungsbefehl zur Ventilöffnung aus dem Anwendungsszenario, treten sporadisch und i.d.R. äußerst selten auf. Damit die für sie geforderte maximale Verzögerung garantiert werden kann, müssen beim Einsatz exklusiver Reservierungen allerdings trotzdem sehr häufig reservierte Slots eingeplant werden, auch wenn diese nur sehr selten genutzt werden. Im Beispiel des Steuerungsbefehls zur Ventilöffnung wird ein minimales Ereignis-Eintrittsintervall Int_{min} von 30 s gefordert, die maximale Übertragungsverzögerung beträgt allerdings 1 s. Es müssten also jede Sekunde ausreichend Slots reserviert werden, obwohl maximal alle 30 s eine Übertragung stattfindet.

[5] Dem Network-Manager selbst werden zunächst mehr Reservierungen zugeteilt als anderen Knoten, da er zu Beginn besonders viele Rahmen senden muss.

Um eine effizientere Unterstützung für ereignisgesteuerte Datenströme bieten zu können, wird neben exklusiven Reservierungen auch *Mode-Based Scheduling with Fast Mode-Signaling* [BGK13] unterstützt. Das Verfahren ermöglicht einen kontrollierten Wettbewerb um Zeitslots, der anhand von Prioritäten deterministisch aufgelöst wird. Anstatt, dass ein Bereich von Mikro-Slots exklusiv für einen Knoten reserviert wird, kann er mit Mode-Based Scheduling für mehrere Knoten reserviert werden. Am Wettbewerb dürfen dabei nur die vorgesehenen Knoten teilnehmen, deren Prioritäten zum Zeitpunkt der Reservierung festgelegt werden. Die Auflösung des Wettbewerbs (Fast Mode-Signaling) erfolgt im Single-Hop-Bereich über unterschiedliche Backoff-Zeiten und kann über mehrere Hops mit Hilfe des *Arbitrating and Cooperative Transfer Protocol (ACTP)* [CGR12] erfolgen.
Mit Mode-Based Scheduling können Slot-Bereiche, die für ereignisbasierten Verkehr reserviert werden müssen, von anderen Knoten genutzt werden, wenn das Ereignis nicht eingetreten ist. So könnte beispielsweise der Steuerungsbefehl zur Ventilöffnung mit hoher Priorität und ein zusätzlicher Temperatur-Sensor-Wert zum Erreichen der bevorzugten Dienstgüte mit niedriger Priorität kombiniert werden. Solange das Ereignis nicht auftritt, kann die bevorzugte Dienstgüte erreicht werden, tritt das Ereignis auf, wird die Basis-Dienstgüte durch andere exklusive Reservierungen sichergestellt.

4.2.2.5 Wettbewerbsbasierter Verkehr

Neben exklusiven Reservierungen und Mode-Based Scheduling ist es außerdem möglich, dass Mikro-Slot-Bereiche für beliebigen Wettbewerb freigegeben werden. Wettbewerbsbasierte Mikro-Slot-Bereiche müssen entweder im gesamten Netzwerk zeitgleich auf demselben Kanal gelten, oder es muss über SDMA/FDMA sichergestellt sein, dass es nicht zu Interferenzen mit exklusiven Reservierungen oder Mode-Based Scheduling kommt. In wettbewerbsbasierten Bereichen dürfen alle Knoten senden, nachdem sie das Medium per Carrier Sense als frei erkannt haben (CSMA). Kollisionen sind daher nicht ausgeschlossen. Der Einsatz von CSMA/CA dient zur Verringerung der Wahrscheinlichkeit andauernder Kollisionen. Damit batteriebetriebene Knoten nicht unnötig stark belastet werden, wird von ihnen nicht verlangt, dass sie in wettbewerbsbasierten Bereichen im Empfangsmodus sind.
Wettbewerbsbasierte Bereiche sind nötig, um neuen Knoten die Möglichkeit zu bieten, sich während des Betriebs am Netzwerk anzumelden. Außerdem kann ein Knoten in wettbewerbsbasierten Bereichen neue Reservierungen anfordern, wenn beispielsweise der Nachbar-Knoten ausgefallen ist, über den er bisher seine Pakete gesendet hat. Auch um mobile Knoten im Netzwerk effizient zu integrieren, bieten wettbewerbsbasierte Bereiche eine Möglichkeit.

4.2.2.6 Acknowledgement und Neuübertragung

Auf MAC-Ebene werden bestätigte Rahmen mit Acknowledgement (ACK) und unbestätigte Rahmen (z.B. Broadcast-Rahmen) unterstützt. Außerdem kann für jeden Rahmen angegeben werden, ob er vom MAC Layer bei ausbleibendem ACK neu übertragen oder verworfen werden soll. Neuübertragungen erfolgen i.d.R. nicht sofort, sondern erst dann, wenn der Knoten die nächste Erlaubnis zum Senden erhält.

4.2.2.7 Lokaler Multicast

Da längere Reservierungen flexibel möglich sind, kann der Duocast-Ansatz von ISA 100.11a auf mehrere Acknowledgements zu einem (lokalen) *Multicast* verallgemeinert werden, bei dem eine

beliebige Anzahl Empfänger den Empfang eines Rahmens per Acknowledgement bestätigt. Damit die Anzahl der benötigten Mikro-Slots bestimmt und die Reihenfolge, in der die Acknowledgements gesendet werden, festgelegt werden kann, muss zum Zeitpunkt der Reservierung allerdings bekannt sein, welche und wie viele Empfänger zu einem Multicast gehören. Durch die Unterstützung lokaler Multicasts mit Acknowledgement müssen Knoten, die einen Rahmen an mehrere Nachbarn senden, diesen nicht mehrfach übertragen. So können Übertragungen eingespart werden, was eine effizientere Nutzung des Mediums und Energieeinsparung zur Folge hat.

4.2.2.8 Prüfsumme

Um verfälschte Rahmen zu erkennen, wird die IEEE 802.15.4-Prüfsumme ausgewertet. Ist die Prüfsumme nicht korrekt, wird der Rahmen verworfen und kein Acknowledgement gesendet. Daraufhin wird eine Verfälschung genau wie Verlust behandelt, was je nach Anforderungen des Datenstroms zu einer Neuübertragung des Rahmens führen kann.

4.2.2.9 Duplikateliminierung

Rahmen werden vom Sender mit einer Sequenz-Nummer versehen, um Duplikate zu erkennen. Wird ein Rahmen doppelt empfangen, wird der zweite Empfang mit einem Acknowledgement bestätigt, aber nicht an die höheren Layer weitergereicht.

4.2.2.10 Channel Hopping

Um eine möglichst gute Zuverlässigkeit auch bei Störungen des Mediums zu erreichen und die Wahrscheinlichkeit von Verlusten zu verringern, wird ähnlich wie bei WirelessHART und ISA 100.11a Channel Hopping eingesetzt. Nachdem ein Knoten eine Reservierung beim Network-Manager angefragt hat, sendet dieser dem Knoten die Reservierungsbestätigung und das für diese Reservierung gültige Sprungmuster.
Ist auf einem Kanal wiederholt keine zuverlässige Übertragung möglich, meldet der Knoten dies dem Network-Manager. Falls möglich, weist dieser dem Knoten daraufhin eine neue Reservierung mit einem Sprungmuster zu, welches die problematische Frequenz nicht enthält. Im Gegensatz zu WirelessHART, welches nur ein statisch konfigurierbares Channel Blacklisting gestattet, erlaubt dieser Ansatz ein *automatisches dynamisches Channel Blacklisting*. Dies verbessert insbesondere das Zusammenspiel des Kommunikationssystems mit anderen Systemen wie WLAN. Bei ISA 100.11a ist ebenfalls ein automatisches Channel Blacklisting möglich, allerdings bleiben dort die reservierten Slots auf den problematischen Kanälen ungenutzt[6]. Indem der Network-Manager dynamisch neue Reservierungen ohne den problematischen Kanal vergibt, führt dies im hier verfolgten Ansatz nicht dazu, dass reservierte Slots dauerhaft unbenutzt bleiben. So kann die Zuverlässigkeit verbessert und das Medium effizienter genutzt werden.

4.2.2.11 Sicherheit

Genau wie bei WirelessHART und ISA 100.11a wird bereits auf MAC-Ebene sämtlicher Verkehr mit einem *Network-Key* verschlüsselt. Zum Beitritt eines Knotens zum Netzwerk wird ein *Join-*

[6] Der System Manager wird bei ISA 100.11a in regelmäßigen Reports über die Link-Qualität informiert und kann bei Bedarf problematische Kanäle blockieren und Routen anpassen. Die genaue Funktionsweise des System Managers wird allerdings nicht spezifiziert.

Key benötigt, daraufhin bekommt der Knoten den aktuell gültigen Network-Key mitgeteilt, mit dem er alle folgenden Übertragungen absichert. Der Network-Key wird vom Network-Manager in regelmäßigen Abständen ausgetauscht und den anderen Knoten mitgeteilt. Die Verschlüsselung auf MAC-Ebene erfolgt wie bei WirelessHART und ISA 100.11a symmetrisch mittels AES.

4.2.2.12 Zeitsynchronisation

Basierend auf der BBS-basierten Ticksynchronisation erfolgt eine netzweite Zeitsynchronisation. Der Network-Manager, der ebenfalls als Masterknoten für die Ticksynchronisation dient, sendet dazu in jedem Super-Slot einmal seine lokale Zeit zum Zeitpunkt des Super-Ticks. Diese Information wird unverändert netzweit weitergeleitet. Da allen Knoten die Dauer eines Makro-Ticks durch statische Konfiguration bekannt ist, entspricht der maximale Clock-Offset dem maximalen Tick-Offset des BBS-Ticks und ist damit deterministisch begrenzt. Die Zeit-Information kann von den höheren Ebenen abgefragt und beispielsweise genutzt werden, um Sensorwerte mit Zeitstempeln zu versehen.

4.2.2.13 Deterministische Arbitrierung und Werteübertragung

Der MAC Layer bietet darüber hinaus Unterstützung für eine deterministische netzweite bzw. n-hop Arbitrierung und Werteübertragung per ACTP [CGR12]. Obere Ebenen können dies nutzen, um kurze Daten netzweit zu übertragen oder eine netzweite Arbitrierung durchzuführen. Diese Funktionalität wird u.a. bei der Ermittlung des Netzzustandes (s. Kapitel 4.2.3.2) genutzt. Wenn die Übertragung netzweit erfolgen soll, müssen für die Nutzung von ACTP netzweit zeitgleich Slots dafür reserviert sein.

4.2.3 Network Layer

Die Reichweite von IEEE 802.15.4 reicht mit üblicherweise ca. 30 m nicht aus, um Produktionsanlagen wie im Anwendungsszenario mit einer Single-Hop-Topologie abzudecken (s. Kapitel 2.1). Daher müssen Pakete u.U. vom Start- über mehrere Zwischenknoten hinweg zum Zielknoten weitergeleitet werden. Diese Weiterleitung ist Aufgabe des Network Layers. Hierbei muss entschieden werden, ob alle Knoten oder nur bestimmte Knoten Pakete weiterleiten können, ob also eine Mesh-Topologie wie bei WirelessHART oder eine Stern-Mesh-Topologie wie bei ISA 100.11a verwendet wird. Weiterhin muss der Network Layer entscheiden, an welchen Knoten ein bestimmtes Paket weitergeleitet werden muss. Um diese Entscheidung treffen zu können, ist ein Routing-Verfahren nötig, welches geeignete Pfade durch das Netzwerk bestimmt. Es ist nicht Ziel dieses Konzepts, zu beschreiben, wie dieses Routing im Detail umgesetzt wird. Stattdessen werden die Anforderungen an das Routing-Verfahren definiert und einige prinzipielle Alternativen verglichen und ausgewählt. Neben dem Routing übernimmt der Network Layer auch Sicherheits-Funktionalität wie z.B. Ende-zu-Ende-Verschlüsselung. Hierauf wird auch nur oberflächlich eingegangen.

4.2.3.1 Topologien

In Kapitel 2.5 wurden unterschiedliche Arten von Topologien vorgestellt, die ein Multi-Hop-Netzwerk bilden kann. Hier werden die Vor- und Nachteile der unterschiedlichen Topologien untersucht und jene gewählt, die am besten zu den Anforderungen passt.

Bei der *Mesh-Topologie* (s. Abb. 2.3) sind alle Knoten gleichwertig: Jeder Knoten ist mit jedem Knoten in seiner Kommunikationsreichweite verbunden und muss für andere Knoten Pakete weiterleiten. Dadurch kann jede mögliche Verbindung zwischen zwei Knoten genutzt werden, was bedeutet, dass die kürzest möglichen Routen verwendet werden können. Dieser Vorteil kann zu kürzeren Verzögerungen und damit einer besseren Performanz führen.
Da sämtliche vorhandenen Verbindungen genutzt werden können, gibt es außerdem in einem Mesh-Netzwerk meist alternative Routen, wenn ein Knoten ausfällt und Pakete über andere Knoten übertragen werden sollen. Dieser Vorteil der Mesh-Topologie trägt zu einer höheren Zuverlässigkeit des Kommunikationssystems bei.

Bei der *Stern-Mesh-Topologie* (s. Abb. 2.4) werden die Knoten in zwei Gruppen aufgeteilt: Die sogenannten *Router* müssen von anderen Knoten Pakete annehmen und diese weiterleiten. Endgeräte müssen dagegen keine Pakete für andere Knoten weiterleiten. Da das Weiterleiten von Paketen Energie kostet, ist es sinnvoll, batteriebetriebene Knoten von dieser Aufgabe zu entlasten. Da eine der Anforderungen ist, dass der Batteriebetrieb einzelner Knoten möglich sein soll (s. Kapitel 2.8), ist dies ein entscheidender Vorteil. Nachteilig ist, dass nicht alle möglichen Verbindungen nutzbar sind. Dies kann dazu führen, dass Routen länger sind und im Falle eines Knotenausfalls weniger alternative Routen vorhanden sind. Allerdings wird hier davon ausgegangen, dass ausschließlich Knoten als Endgerät konfiguriert werden, die auf Grund ihrer Energieversorgung als Router ungeeignet sind. Um auch bei einer Stern-Mesh-Topologie möglichst kurze Routen und eine gute Zuverlässigkeit zu erreichen, müssen lediglich genügend Router in der Nähe von Endgeräten platziert werden. Bei einem drahtlosen Netzwerk in einer Produktionsanlage ist es problemlos möglich, gezielt Router zu platzieren, sodass sich keine Nachteile im Vergleich mit einer Mesh-Topologie ergeben. Daher wird hier wie bei ISA 100.11a eine Stern-Mesh-Topologie verwendet.

4.2.3.2 Ermittlung des Netzzustands

Zur Routen-Berechnung muss der Network Layer Informationen über die Topologie des Netzwerks haben. Wäre diese Information statisch konfiguriert, könnten nicht – wie in den Anforderungen gefordert – während des Betriebs Knoten zum Netzwerk dazukommen (s. Kapitel 2.6). Daher soll das System den Netzzustand beim Start selbstständig ermitteln und während des Betriebs aktualisieren, wenn Topologie-Änderungen erkannt wurden.

Für die initiale Topologie-Bestimmung wird ein spezielles Protokoll zur Ermittlung des Netzzustands verwendet [Kra13]. Das Protokoll testet sämtliche Links auf ihre Empfangsstärke und teilt sie in mehrere Gruppen ein:

- Ein *Kommunikations-Link* von einem Knoten A zu einem Knoten B besteht dann, wenn B Pakete von A (direkt) korrekt empfangen kann, sofern kein anderer Knoten in Interferenzreichweite zeitgleich sendet.
- Ein *Interferenz-Link* von einem Knoten A zu einem Knoten B besteht dann, wenn das korrekte Empfangen von Paketen durch B gestört wird, wenn A zeitgleich sendet. Ein Kommunikations-Link ist immer auch ein Interferenz-Link.
- Ein *Sensing-Link* von einem Knoten A zu einem Knoten B besteht dann, wenn B Übertragungsenergie von A erkennen kann. Ein Interferenz-Link ist immer auch ein Sensing-Link.

Das Wissen über Interferenz-Links ermöglicht es, effizient SDMA einzusetzen, sodass räumlich getrennte Knoten zeitgleich auf dem gleichen Kanal senden können.

Da das Protokoll selbst bereits die netzweite Verteilung der ermittelten Topologie-Information übernimmt, kann davon ausgegangen werden, dass jeder Knoten über dieses Wissen verfügt. Dies ermöglicht es beispielsweise, dass jeder Knoten Routen oder Entfernungen zu anderen Knoten berechnen kann[7].

Werden Änderungen an der Topologie bemerkt, wie beispielsweise ausgefallene Knoten oder gebrochene Links, so wird diese Information zum Network-Manager gesendet, der diese Information im Netzwerk verteilt.

4.2.3.3 Routing

Die in Kapitel 2 formulierten Anforderungen stellen einige Herausforderungen an das Routing: Um Dienstgüte garantieren zu können (s. Kapitel 2.4), muss auch das Routing Dienstgüte-Unterstützung bieten. Das bedeutet, dass bei der Suche einer Route nicht immer die kürzeste Route gesucht ist, sondern eine Route, welche eine gegebene Dienstgüte-Anforderung erfüllt. Nur so können beispielsweise die Anforderungen an die maximale Übertragungsverzögerung (s. Kapitel 2.2) sichergestellt werden.

Eine weitere wichtige Anforderung an das Routing ist, dass neben stationären Knoten auch Unterstützung für mobile Knoten, wie beispielsweise Roboter, vorhanden ist (s. Kapitel 2.6). Da allerdings nur ein kleiner Teil der Knoten mobil ist, sollte das Routing eine Unterscheidung zwischen stationären und mobilen Knoten vorsehen. So ist es beispielsweise denkbar, Routen zwischen stationären Knoten proaktiv vorzuberechnen, während Routen zu mobilen Knoten bei Bedarf reaktiv bestimmt werden. In Kapitel 4.2.3.4 wird eine Möglichkeit skizziert, wie eine Unterstützung für mobile Knoten aussehen könnte.

Eine wichtige Voraussetzung für das Routing ist, dass sämtlichen Knoten die vollständige Topologie der stationären Knoten bekannt ist (s. Kapitel 4.2.3.2). Anhand dieser Information ist es beispielsweise möglich, dass die Knoten lokal Routen berechnen. Allerdings kann nicht davon ausgegangen werden, dass allen Knoten sämtliche Kommunikations-Ressourcen im Netzwerk bekannt sind. Die Berechnung einer Route, welche eine bestimmte Dienstgüte erfüllt, ist daher nicht ausschließlich lokal möglich.

Ähnlich wie das Graph-Routing von WirelessHART und ISA 100.11a (s. Kapitel 3.2.3) sollte das Routing die Übertragung auf einer alternativen Route ermöglichen, wenn beispielsweise ein Knoten ausfällt. So kann die Zuverlässigkeit des Netzwerks verbessert werden.

Das Anwendungsszenario beschreibt auch, dass ein Dienst eines Knotens (z.B. der Wert eines Temperatur-Sensors) von mehreren Knoten (z.B. Reglern) abonniert werden kann (s. Kapitel 2.6). Um dies effizient zu unterstützen, wäre es wünschenswert, wenn über die lokale Multicast-Unterstützung auf MAC-Ebene hinaus auch eine netzweite Multicast-Unterstützung auf Ebene des Network Layers möglich ist. Dies bedeutet für das Routing, dass nicht nur Routen von einem Knoten zu einem anderen, sondern auch Routen von einem Knoten zu einer Menge von anderen Knoten berechnet werden können.

Wie genau ein effizientes Routing-Verfahren unter den gegebenen Anforderungen aussehen kann, ist eine komplexe Fragestellung, die im Rahmen einer anderen Arbeit untersucht werden wird. Standard-Verfahren wie DSDV oder AODV sind bei den gegebenen Anforderungen nicht geeignet: Sie zielen in erster Linie darauf ab, die für das Routing benötigte Topologie-Information des Netzwerks zu ermitteln bzw. auszutauschen. Da hier davon ausgegangen werden kann, dass die Topologie allen Knoten bekannt ist, ist dies nicht Aufgabe des Routing-

[7] Details zum Routing werden in Kapitel 4.2.3.3 erläutert.

Verfahrens. Dagegen müssen andere Probleme wie die Dienstgüte-Unterstützung oder die Unterstützung mobiler Knoten gelöst werden.

4.2.3.4 Mobile Knoten

Die Anforderungen verlangen, dass neben stationären Knoten auch mobile Knoten unterstützt werden sollen (s. Kapitel 2.6). Wie dies im Detail realisiert wird, ist nicht Gegenstand dieser Arbeit. Im Folgenden wird allerdings kurz skizziert, wie eine mögliche Umsetzung aussehen könnte.

Bei der initialen Topologie-Bestimmung sind mobile Knoten nicht beteiligt und die Position mobiler Knoten kann nicht als allen Knoten bekannt angenommen werden. Wenn ein mobiler Knoten dem Netz beitritt, wählt er einen stationären, nicht batteriebetriebenen Knoten in seiner Reichweite als *Home Agent* aus. Dazu senden diese Knoten in regelmäßigen Abständen *Beacons*, anhand derer sie von mobilen Knoten erkannt werden können. Bewegt sich der mobile Knoten, ändert sich sein Home Agent nicht; selbst dann, wenn er sich aus dessen Reichweite hinaus bewegt. In regelmäßigen Abständen teilt der mobile Knoten seinem Home Agent mit, welche stationären Knoten gerade in seiner Reichweite sind. Dazu sendet er das Paket an einen der stationären Knoten in Reichweite, der die weitere Route zum Home Agent berechnet. Indem der Home Agent stets die stationären Knoten in Reichweite des mobilen Knotens kennt, kann er immer eine Route zu diesem berechnen. Home Agents machen dem Network-Manager bekannt, für welche mobilen Knoten sie Home Agent sind, und der Network Manager verteilt daraufhin diese Information im Netz. Möchte ein Knoten ein Paket an einen mobilen Knoten senden, so kann er dies tun, indem er es an den Home Agent des mobilen Knotens sendet, der es an den mobilen Knoten weiterleitet. Sendet ein Knoten häufig Pakete an einen mobilen Knoten, so kann er außerdem bei dessen Home Agent einen Dienst abonnieren, der ihn stets über Positionsänderungen des mobilen Knotens informiert. Mit der durch diesen Dienst erhaltenen Position kann der Knoten eine Route berechnen, über die er Pakete direkt an den mobilen Knoten sendet, ohne sie über den Home Agent verschicken zu müssen.

4.2.3.5 Segmentierung

Der Network Layer nimmt Pakete entgegen, die größer sein können als die maximale Rahmengröße auf MAC-Ebene. Er segmentiert dazu automatisch große Pakete auf Senderseite und setzt sie auf Empfängerseite wieder zusammen.

Die Anforderungen verlangen, dass QoS-Garantien eingehalten werden (s. Kapitel 2.4). Diese können aber nicht für beliebig große Pakete garantiert werden. Mit zunehmender Segmentierung steigt beispielsweise die Übertragungszeit an. Aus diesem Grund muss beim Anfordern einer Reservierung die maximale Paketgröße des Datenstroms angegeben werden. Daraus berechnet der Network Layer die maximal nötige Segmentierung und damit die maximal benötigte Anzahl Rahmen auf MAC-Ebene. Daraus ergibt sich, wie viele Slots reserviert werden müssen, um die geforderten QoS-Anforderungen garantieren zu können.

4.2.3.6 Sicherheit

Wie in Kapitel 2.7 erläutert wurde, muss die Sicherheit des drahtlosen Kommunikationssystems gewährleistet sein. Um die Vertraulichkeit der übertragenen Nachrichten sicherzustellen, wird neben der Hop-zu-Hop-Verschlüsselung auf MAC-Ebene vom Network Layer optional

eine *Ende-zu-Ende-Verschlüsselung* unterstützt. Der Network Manager generiert dazu für jedes Knoten-Paar einen eigenen Schlüssel, der *Path Key* genannt wird, und übermittelt diesen an die beiden Knoten, deren Kommunikation mit dem Schlüssel abgesichert wird. Die erstmalige Übermittlung des Schlüssels wird über die Verschlüsselung auf MAC-Ebene gesichert. Alle weiteren Übertragungen werden mit den Path Keys verschlüsselt.
Möchte ein Knoten eine verschlüsselte Multicast-Verbindung nutzen, muss er dazu einen Path Key beim Network Manager anfordern. Dabei teilt er dem Network Manager die Empfänger der Multicast-Übertragung mit. Der Network Manager generiert einen neuen Path Key und überträgt ihn an die Empfänger und den Absender, der ihn ab dann für Multicast-Übertragungen nutzen kann.

Neben der symmetrischen Verschlüsselung der Übertragung unterstützt der Network Layer auch die digitale Signatur von Nachrichten. Dies dient der geforderten Sicherstellung von Authentizität und Integrität der Nachrichten (s. Kapitel 2.4). Dazu generiert der Knoten selbst ein Schlüsselpaar aus privatem und öffentlichem Schlüssel und sendet den öffentlichen Schlüssel an den Network Manager. Andere Knoten können den öffentlichen Schlüssel beim Network Manager abfragen, um beispielsweise eine Signatur zu überprüfen.

Genau wie die Hop-zu-Hop-Verschlüsselung auf MAC-Ebene verwenden die Verschlüsselung und die digitale Signatur des Network Layers das AES-Verfahren.

4.2.4 Middleware Layer

Der Middleware Layer stellt der Applikation eine dienstorientierte Schnittstelle zum Kommunikationssystem zur Verfügung, welche insbesondere die in Kapitel 2.6 geforderten Anforderungen an die Flexibilität des Systems erfüllt. Der vor allem aus dem Umfeld von Informationssystemen als *Service-Oriented-Architecture* (SOA) [Erl05] bekannte Ansatz, Funktionalität in Form von Diensten bereitzustellen, kommt hier in einem drahtlosen Kommunikationssystem zum Einsatz.

Ein *Dienst* ist eine in sich abgeschlossene Teilfunktionalität der Applikation. Ein typisches Beispiel im Kontext von Produktions-Netzwerken ist das Auslesen eines Sensorwertes oder das Steuern eines Aktuators. Der eigentliche Dienst wird von der Applikation implementiert, während der Middleware Layer Funktionalität zum Auffinden und Aufrufen von Diensten bereitstellt. Ein Knoten kann einen oder mehrere Dienste anbieten, die dann über das drahtlose Netzwerk von anderen Knoten verwendet werden können. Zur Kommunikation mit einem Dienst werden drei unterschiedliche *Message-Exchange-Patterns* (MEPs) unterstützt (vgl. [Erl05]):

- *Request/Response:* Ein Dienst wird aufgerufen (Request) und gibt ein Ergebnis als Antwort an den Aufrufer zurück (Response). Beispielsweise fragt ein Knoten einen Sensorwert an und erhält als Antwort den aktuellen Wert des Sensors. Dieses Muster entspricht dem entfernten Aufruf einer Funktion mit Rückgabewert (Remote Procedure Call).
- *Fire and Forget:* Ein Dienst wird aufgerufen, allerdings wird keine Antwort zurückgegeben. Beispielsweise sendet ein Regler einen Regelungsbefehl an einen Aktuator, ohne dass eine Antwort zurück gegeben wird. Dieses Muster entspricht dem entfernten Aufruf einer Funktion ohne Rückgabewert (Remote Procedure Call).
- *Publish/Subscribe:* Ein Dienst wird vom Dienstanbieter im Netzwerk bekannt gemacht (Publish) und kann daraufhin von anderen Knoten im Netzwerk abonniert werden (Subscribe). Beispielsweise kann der Wert eines Sensors abonniert werden, sodass der Sensorknoten in regelmäßigen Abständen den aktuellen Sensorwert an alle Abonnenten sendet, ohne dass diese ihn jedes Mal abfragen müssen.

Es gibt weiterhin zwei unterschiedliche Arten von Publish/Subscribe-Diensten:

- Dienste, welche *periodisch* in einem festen Intervall eine Nachricht senden. Beispielsweise sendet ein Sensorknoten i.d. R. periodisch den aktuellen Sensorwert.
- Dienste, welche *ereignisgetriggert* immer dann eine Nachricht senden, wenn ein bestimmtes Ereignis eintritt. Beispielsweise sendet eine Lichtschranke immer dann eine Nachricht, wenn sie unterbrochen wird.

Ein Knoten, der einen Dienst anbietet (*Service Provider*), macht diesen bei der *Service Registry* bekannt (Publish, s. Abb. 4.5). Die Service Registry ist konzeptionell eine Datenbank, in der die im Netzwerk verfügbaren Dienste gespeichert sind. Ein Knoten, der einen Dienst nutzen möchte (*Service Requestor*), kann eine Anfrage an die Service Registry senden, um nach einem Dienst zu suchen (s. Abb. 4.5). Die Service Registry antwortet mit einer Liste verfügbarer Dienste. Anhand dieser Liste weiß der Service Requestor, welche Knoten die gewünschten Dienste anbieten und welches MEP jeweils verwendet wird. Entsprechend dieses MEPs interagiert er nun mit dem Service Provider: Handelt es sich um einen Dienst, der nach dem Request/Response-MEP aufgerufen wird, sendet er einen Aufruf und bekommt eine Antwort zurück. Kommt Fire and Forget zum Einsatz, bleibt der Aufruf des Dienstes dagegen unbeantwortet. Verwendet der Dienst das Publish/Subscribe-Verfahren, sendet der Service Requestor eine Subscribe-Nachricht an den Service Provider, um den Dienst zu abonnieren. Dieser wiederum sendet daraufhin in regelmäßigen Abständen oder immer, wenn ein bestimmtes Ereignis auftritt, eine Nachricht an den Service Requestor, bis dieser den Dienst abbestellt.

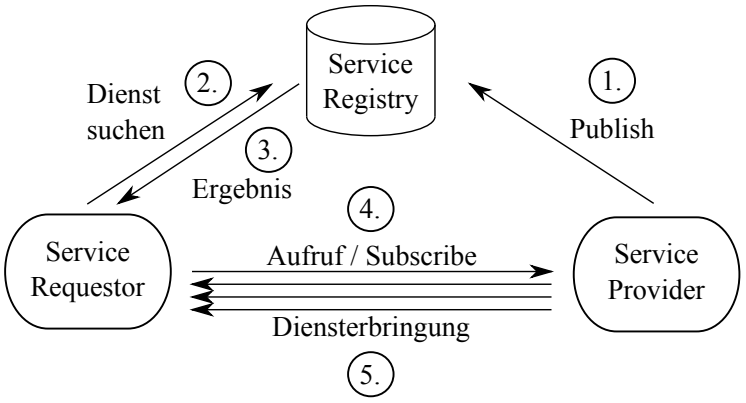


Abbildung 4.5.: Interaktion zwischen Service Registry, Service Requestor und Service Provider (nach [Erl05])

Der hier verfolgte dienstorientierte Ansatz bietet vor allem Vorteile gegenüber dem Kommando-basierten Ansatz von WirelessHART und dem objektorientierten von ISA 100.11a, weil er Service Requestor und Service Provider voneinander entkoppelt. Der Service Requestor muss nicht wissen, welcher Knoten beispielsweise den Sensorwert eines bestimmten Sensors bereitstellt. Er benötigt lediglich die *Dienstkennung*, um in der Service Registry nachzuschlagen, welcher Knoten den Sensorwert liefern kann. Auf diese Weise ist es beispielsweise problemlos möglich, einen Knoten während des Betriebs auszutauschen, wie es in Kapitel 2.6 gefordert wurde.

Um, wie in Kapitel 2.4 verlangt wurde, Dienstgüte-Garantien anbieten zu können, muss auch der Middleware Layer Dienstgüte-Unterstützung bieten. Beim Publizieren eines Dienstes können Kenndaten der unterstützten Dienstgüte angegeben werden, wie etwa das kleinst mögliche Intervall, in dem ein Sensorwert verfügbar gemacht werden kann. Beim Suchen eines Dienstes können ebenfalls Dienstgüteanforderungen angegeben werden, sodass nicht den Anforderun-

gen entsprechende Dienste vorgefiltert werden. Wird ein Dienst abonniert, kann die benötigte Dienstgüte (z.B. eine maximale Übertragungsverzögerung) gefordert werden. Der Middleware Layer nutzt die Dienstgüte-Unterstützung der darunter liegenden Schichten, um zu überprüfen, ob die geforderte Dienstgüte eingehalten werden kann. Ist dies der Fall, stellt er sicher, dass das Abonnement mit der geforderten Dienstgüte erbracht wird. Kann die geforderte Dienstgüte nicht garantiert werden, wird das Abonnement abgelehnt und muss ggf. mit einer geringeren Dienstgüte oder von einem anderen Knoten, der den gleichen Dienst anbietet, neu abonniert werden.
In Kapitel 5 wird genauer auf den Middleware Layer und dessen Umsetzung eingegangen. Dabei wird insbesondere detailliert beschrieben, wie die Service Registry umgesetzt ist.

4.2.5 Zusammenfassung

Das hier entwickelte Konzept zeigt auf den unteren Layern Gemeinsamkeiten mit WirelessHART und ISA 100.11a: So wird mit IEEE 802.15.4 der gleiche Physical Layer eingesetzt und der MAC Layer basiert ebenfalls auf einer Kombination aus TDMA mit SDMA und FDMA. Allerdings zeigt schon der MAC Layer entscheidende Unterschiede. So wird die Zeitsynchronisation nicht anhand von Nutzrahmen durchgeführt, sondern BBS zur Synchronisation eingesetzt. Dies macht die Zeitsynchronisation unabhängig von den oberen Layern und gewährleistet einen deterministisch beschränkten Offset. Die exklusiven Reservierungen sind durch ihre variable Länge flexibler als bei WirelessHART und ISA 100.11a, was eine bessere Ausnutzung des Mediums ermöglicht. Mit Mode Based Scheduling bietet der MAC Layer darüber hinaus Unterstützung für deterministischen Wettbewerb. Zusätzlich sind ähnlich wie beim Slow Hopping von ISA 100.11a wettbewerbsbasierte Bereiche vorgesehen. Reservierungen müssen nicht statisch konfiguriert werden, sondern können während des Betriebs dynamisch angefordert werden. Dies erlaubt es auf Anwendungsebene, Dienste flexibel bei Bedarf zu abonnieren und abzubestellen. Die automatische Ermittlung der Topologie macht das System flexibler und ermöglicht effizientes SDMA. Das Routing bietet wie gefordert Unterstützung für mobile Knoten sowie Dienstgüte. Ähnlich wie bei WirelessHART und ISA 100.11a wird die Zuverlässigkeit verbessert, indem z.B. bei Knotenausfall Neuübertragungen auf alternativen Routen stattfinden. Indem zwischen Routern und Endgeräten unterschieden wird, werden batteriebetriebene Knoten besser als bei WirelessHART unterstützt. Neben der Hop-zu-Hop-Verschlüsselung auf MAC-Ebene bietet der Network Layer Ende-zu-Ende-Verschlüsselung sowie Unterstützung für digitale Signaturen. Der Middleware Layer stellt mit seinem dienstorientierten Ansatz eine besonders flexible Schnittstelle zur Anwendung dar.
Die Entscheidungen, die in diesem Konzept getroffen wurden, sind durch die in Kapitel 2 formulierten Anforderungen motiviert. In Tab. 4.1 ist zusammenfassend dargestellt, welche Anforderungen an Performanz, Zuverlässigkeit, Skalierbarkeit, Flexibilität, Sicherheit und Ressourcen-Effizienz sowie funktionale Anforderungen durch die Entscheidungen auf den unterschiedlichen Layern des Protokolls erfüllt werden bzw. die Erfüllung unterstützen. So wird beispielsweise auf dem MAC Layer TDMA eingesetzt (s. Kapitel 4.2.2), um, wie in den Anforderungen in Kapitel 2.2 verlangt, Übertragungsverzögerungen garantieren zu können. Zusätzlich dazu nutzt der MAC Layer SDMA und FDMA, um die Übertragungsrate des Netzwerks zu verbessern. Die in Kapitel 2.2 aufgestellte Forderung einer Brutto-Übertragungsrate von mindestens 200 kbit/s wird schon durch den Physical Layer von IEEE 802.15.4 erfüllt, der 250 kbit/s ermöglicht. SDMA und FDMA sind daher nicht notwendig, um die Anforderung zu erfüllen, verbessern aber die Erfüllung der Anforderung.

Tabelle 4.1.: Anforderungen und ausgewählte Lösungen nach Layern. Legende: erfüllt Anforderung / *unterstützt Erfüllung der Anforderung*

	Physical Layer	MAC Layer	Network Layer	Middleware Layer
Performanz				
Übertragungsrate	IEEE 802.15.4	*FDMA, SDMA (zusätzlich)*	*Topologie-Bestimmung*	
Übertragungsverzögerung		TDMA (exkl. Reservierungen, *Mode Based*), (Ticksynchronisation)	QoS-Routing	Dienste unterst. QoS-Anf. für periodische und ereignis-gesteuerte Datenströme
Zuverlässigkeit				
Verluste		Exklusive Reservierungen, *Channel Hopping (+Blacklisting)*, ACK + Neuübertragung (je nach Anforderung)	*Neuübertragung auf alternativer Route bei Knotenausfall*	
Verfälschungen		IEEE 802.15.4 Prüfsumme, Neuübertragung (je nach Anf.)		
Umordnung, Duplikateliminierung		Sequenz-Nummern		*Zeitstempel*
Skalierbarkeit				
Knotenanzahl		Adressraumgröße mind. 1 Byte		*Dienste-Kennungen*
Topologien		SDMA, Multi-Hop-Arbitrierung	Routing	
Flexibilität				
Dynamische Dienste		Dyn. Superslotkonfiguration		Dienste-Registry
Knotenaustausch			Periodische Routen Aktualisier-ung, Topologie-Bestimmung	
Mobile Knoten			Routing unterstützt mobile Knoten (proaktiv/reaktiv -> hybrid)	
Sicherheit				
Vertraulichkeit		Verschlüsselung (Network Key)	Verschlüsselung (Path Keys)	
Integrität, Authentizität			Digitale Signatur	
Verfügbarkeit		*Channel Hopping (+Blacklisting)*, Neuübertragung (je nach Anf.)		
Ressourcen-Effizienz				
Energie-Effizienz	IEEE 802.15.4	Duty Cycling, *Multicast*	*Stern-Mesh-Topologie*, *Multicast*, *Proaktive Routen-Berechnung*	
Rechen- und Speicher Eff.		Geeignete Algorithmen	Geeignete Algorithmen	Geeignete Algorithmen
Größe und Gewicht	Imote 2			
Funktionale Anforderungen				
Zeitsynchronisation		Tick- und Zeitsynchronisation		
Determ. netzw. Arbitrierung und Werteübertragung		ACTP		

5. KAPITEL

Middleware für drahtlose Kommunikation im Produktionsbereich

In Kapitel 4.2.4 wurde der Middleware Layer des entworfenen Protokolls bereits kurz beschrieben. In diesem Kapitel wird der im Rahmen dieser Arbeit implementierte Middleware Layer genauer betrachtet und konzeptionelle Entscheidungen analysiert.

Als Middleware wird hier eine dienstorientierte Schnittstelle der Applikation zum drahtlosen Kommunikationssystem verstanden (s. Abb. 4.2). Ähnlich wie bei RPC (Remote Procedure Call) bzw. RMI (Remote Method Invocation) handelt sich um eine *kommunikationsorientierte Middleware*, welche von der Kommunikation über das Netzwerk abstrahiert. Die Middleware selbst dient dabei der Applikation als Kommunikations-Framework. Darüber hinaus ist eine Service Registry, welche die im System vorhandenen Dienste verwaltet und für Dienstnutzer auffindbar macht (s. Abb. 4.5), Teil der Middleware. Der Fokus dieser Arbeit liegt auf der konzeptionellen Ausarbeitung und Implementierung dieser Service Registry.

Nachdem kurz die Vorteile einer dienstorientierten Middleware-Schicht erläutert werden, beschreibt dieses Kapitel die Architektur des Middleware Layers.

5.1 Vorteile einer dienstorientierten Middleware-Schicht

Verglichen mit dem Kommando-basierten Ansatz von WirelessHART und dem objektorientierten Ansatz von ISA 100.11a bietet der hier verfolgte dienstorientierte Ansatz einige Vorteile:

- Dadurch, dass Dienste in der Service Registry anhand von Dienstkennungen aufgesucht werden können, wird der Dienstnutzer (Service Requestor) vom Dienstanbieter (Service Provider) entkoppelt. Ein Dienst kann daher leicht während des Betriebs auf einen anderen Knoten verschoben oder ein Knoten ausgetauscht werden. Dienstnutzer müssen den Dienst lediglich in der Service Registry erneut nachschlagen und neu abonnieren.
- Durch die Dienstgüte-Unterstützung des Middleware Layers (und der darunter liegenden Layer) kann die Anwendung einen Dienst mit einer gewissen Dienstgüte anfordern und muss sich nicht darum kümmern, wie diese garantiert werden kann.

Neben diesen Vorteilen, welche zur Flexibilität des Systems beitragen, führt der dienststorientierte Ansatz außerdem zu einer klaren Strukturierung der Anwendung und verbessert damit die Wartbarkeit des Systems.

5.2 Architektur des Middleware Layers

Im Folgenden wird detaillierter auf die architektonischen Fragen eingegangen, die sich bei der Entwicklung des Middleware Layers stellen, und dargestellt, welche Lösungen schließlich implementiert wurden. Da die Service Registry hier im Fokus steht, werden die unterschiedlichen konzeptionellen Optionen, eine Service Registry umzusetzen, erörtert. Dabei wird auf Replikation und Verteilung eingegangen und ein Algorithmus vorgestellt, der basierend auf der Topologie des Netzwerks Knoten auswählt, auf welche die Service Registry verteilt wird. Anschließend wird beschrieben, wie ein Dienst der Service Registry bekannt gemacht wird und wie Dienstinformationen verteilt werden. Schlussendlich wird erklärt, wie ein Dienst, der von einem anderen Knoten angeboten wird, in der Service Registry aufgefunden werden kann.

5.2.1 Replikation und Verteilung von Service Registries

Damit die Dienste im Netz aufgefunden werden können, werden sie bei einer Service Registry registriert. Für die Umsetzung dieser Service Registry gibt es mehrere konzeptionelle Optionen bezüglich Replikation und Verteilung. Resultierende Vor- und Nachteile werden hier betrachtet.

5.2.1.1 Replikation

Eine konzeptionell wichtige Entscheidung ist es, ob Informationen über Dienste repliziert werden sollen oder nicht. Wenn die Information über einen Dienst nicht repliziert wird, bedeutet dies, dass sie auf nur einem Knoten gespeichert ist. Dies hat den Vorteil, dass es keine Inkonsistenzen zwischen unterschiedlichen Kopien geben kann. Außerdem muss nur ein Knoten informiert werden, wenn sich die Information ändert. Nachteilig ist dagegen, dass nur dieser eine Knoten die Information anbieten kann, daher der Pfad zu diesem Knoten evtl. sehr lang ist und die Antwort erst mit einer hohen Verzögerung eintrifft. Außerdem ist die Information überhaupt nicht mehr verfügbar, falls der Knoten, auf dem die Information gespeichert ist, ausfällt oder auf Grund von Störungen auf dem Drahtlosmedium nicht erreichbar ist.
Wenn Dienstinformation repliziert wird, bedeutet dies, dass die Information über einen Dienst auf mehrere Knoten kopiert wird. Auf diese Weise bleibt die Information verfügbar, wenn einer der Knoten ausfällt oder nicht mehr erreichbar ist. Werden viele Kopien im Netzwerk verteilt, so führt dies außerdem dazu, dass die Pfade zu Knoten, die über eine Kopie verfügen, kürzer werden. Auf diese Weise wird der Zugriff auf Dienstinformationen schneller. Nachteilig ist, dass es zwischenzeitlich zu Inkonsistenzen zwischen den Kopien kommen kann. Außerdem müssen zusätzliche Nachrichten ausgetauscht werden, um die Replikationen zu aktualisieren.
Als Kompromiss kann der Grad der Replikation bei der implementierten Middleware über einen Parameter konfiguriert werden. Dienstinformation wird im Umkreis von n Hops um den Service Registry-Knoten repliziert, bei dem der Dienst zuerst registriert wurde. Soll keine Replikation stattfinden, kann $n = 0$ gesetzt werden. Indem n mindestens auf den Netzdurchmesser gesetzt wird, kann eine netzweite Replikation erreicht werden. Wird n auf einen Wert zwischen 1 und dem Netzdurchmesser gesetzt, so erfolgt eine lokal begrenzte Replikation. Diese hält den Aufwand der Replikation in Grenzen, da kein Broadcast über das ganze Netz erfolgt, sondern die weitere Übertragung nach n Hops eingestellt werden kann. Trotzdem ist die Information noch verfügbar, falls ein Knoten ausfällt. Geht man davon aus, dass in einem drahtlosen Kommunikationssystem im Produktionsbereich die Sensorknoten und die zugehörigen Regler- und Aktuatorknoten oft räumlich nahe beieinander liegen, so wird der Zugriff auf die Dienstinformation

und damit das Finden und Abonnieren von Diensten in den meisten Fällen durch die lokale Replikation beschleunigt. Auf Grund der genannten Vorteile wurde diese Lösung mit einem konfigurierbarem n gewählt.

5.2.1.2 Verteilung

Eine weitere wichtige konzeptionelle Entscheidung ist, ob die Service Registry zentralisiert oder verteilt umgesetzt werden soll.

Zentralisierte Registry Wird die Service Registry zentral auf einem Knoten eingerichtet, so bedeutet dies, dass alle Dienste bei einem Knoten registriert werden und alle Knoten bei diesem einen Knoten anfragen, wenn sie Dienste suchen. Diese Lösung hat in erster Linie den Vorteil, dass sie einfach umsetzbar ist. Da sämtliche Dienstanfragen und Bekanntmachungen zu diesem einen Knoten gesendet werden müssen, stellt der Knoten bei einem großen Netz mit vielen Anfragen allerdings unter Umständen einen Flaschenhals dar. Zudem können bei einem großen Netz die Pfade zu diesem Knoten lang werden, was zu einer hohen Verzögerung beim Suchen und Registrieren von Diensten führen kann. Da alle Dienstinformationen vollständig am zentralen Knoten gespeichert sein müssen, kann dies außerdem bei einem großen Netzwerk mit vielen Diensten zu Speicherproblemen am zentralen Knoten führen. Insgesamt ist eine zentralisierte Registry vor allem für kleine Netzwerke geeignet, nicht aber für große, da sie nicht skaliert.

Vollständig verteilte Registry Im extremen Gegensatz zur zentralisierten Registry kann die Aufgabe der Service Registry auch auf alle Knoten im Netz verteilt werden. Nachteilig daran ist, dass auch schwache Knoten, die keine feste Energieversorgung und eventuell weniger Speicher besitzen, Dienstinformation speichern und Nachrichten über Dienständerungen empfangen und ggf. weiterleiten müssen. Wie in den Anforderungen in Kapitel 2.8 gefordert, muss allerdings der Batteriebetrieb einzelner Knoten möglich sein. Da eine vollständig verteilte Registry die Lebensdauer batteriebetriebener Knoten deutlich verkürzt, passt diese nicht gut zu den gegebenen Anforderungen.

Partiell verteilte Registry Wenn weder alle Knoten noch ein einziger Knoten als Service Registry agieren, so bedeutet dies, dass eine echte Teilmenge der Knoten die Aufgabe der Service Registry übernimmt. So kann das Problem der vollständig verteilten Registry umgangen werden, bei der Knoten ohne feste Energieversorgung oder mit wenig Speicher auch als Service Registry fungieren müssen. Damit das Nachrichtenaufkommen beim Aufsuchen von Diensten gering ist, sollten möglichst viele Knoten möglichst gleichmäßig in der Topologie verteilt als Service Registry agieren, sodass die Pfade zum nächsten Service Registry-Knoten möglichst kurz sind. Damit der Aufwand beim Bekanntmachen von neuen Diensten oder Dienständerungen gering ist, sollten dagegen möglichst wenige Knoten, die möglichst nah beieinander liegen, als Service Registry dienen. Die Auswahl der Knoten ist daher bei diesem Ansatz eine Herausforderung. Gelingt es, eine geschickte Auswahl der Knoten für die Service Registry zu treffen, so stellt dieser Ansatz aber eine skalierbare Lösung dar, welche sowohl das Nachrichtenaufkommen gering hält als auch berücksichtigt, dass manche Knoten möglichst wenig belastet werden sollen.

Da die partiell verteilte Registry am ehesten zu den gegebenen Anforderungen passt, wurde dieser Ansatz für die Implementierung der Service Registry gewählt. Die Realisierung einer

partiell verteilten Service Registry eröffnet einige weitere Fragestellungen, auf die im Folgenden eingegangen wird.

5.2.2 Auswahl von Knoten für die Service Registry

Da die Registry auf eine Untermenge der Knoten verteilt werden soll, stellt sich die Frage, wie man diese Untermenge auswählt. Ziel ist es, dass ein Algorithmus die Auswahl selbstständig durchführt und dabei gewisse Entwurfsziele beachtet.

5.2.2.1 Problemstellung und Entwurfsziele

Der Algorithmus soll aus der Menge aller stationären Knoten V eine Teilmenge an Knoten V_{SR} auswählen, welche Service Registry-Knoten werden. Ein Knoten, der nicht Service Registry-Knoten wird, bekommt darüber hinaus vom Algorithmus einen Service Registry-Knoten in seiner Nähe zugewiesen, über den er auf die Service Registry zugreifen kann.

Mobile Knoten werden vom Algorithmus nicht berücksichtigt. Auf Grund ihrer Mobilität sind sie prinzipiell als Service Registry weniger geeignet als stationäre Knoten und sollen daher nicht dafür ausgewählt werden. Zum Zugriff auf die Service Registry nutzen sie ihren Home Agent (s. Kapitel 4.2.3.4), der Teil der Service Registry sein muss[8].

Bei der Auswahl der Knoten sollen folgende Ziele verfolgt werden:

Entfernung zu Service Registry-Knoten Jeder Knoten im Netzwerk soll entweder selbst Service Registry-Knoten sein oder einen Service Registry-Knoten als direkten Nachbarn haben, über den er auf die Service Registry zugreifen kann. Damit ist sichergestellt, dass Anfragen an die Service Registry nur über maximal einen Hop übertragen werden müssen und daher schnell bearbeitet werden können[9].

Konnektivität zwischen Service Registry-Knoten Die Service Registry-Knoten müssen häufiger aktuelle Informationen über Dienste untereinander austauschen. Aus diesem Grund soll ein zusammenhängendes Netzwerk aus Knoten gebildet werden, welches alle Service Registry-Knoten miteinander verbindet. Zum Austausch von Informationen innerhalb der Service Registry wird ausschließlich dieses Netzwerk verwendet. Der Algorithmus soll daher neben den Knoten V_{SR}, welche als Service Registry agieren, falls nötig auch zusätzliche Knoten V_{router} auswählen, welche die Service Registry-Knoten miteinander verbinden.

Anzahl ausgewählter Knoten Es sollen unter Beachtung der bisher genannten Ziele möglichst wenig Knoten für die Service Registry ausgewählt werden. Umso weniger Knoten Teil der Service Registry sind, umso weniger Nachrichten sind nötig, um Informationen in der Service Registry zu verteilen oder aufzusuchen. Über das erste Entwurfsziel (Entfernung zu Service Registry-Knoten), welches höher gewichtet wird, ist bereits sichergestellt, dass die Entfernung zur Service Registry für alle Knoten gering ist.

[8] Dies wird sichergestellt, indem mobile Knoten ausschließlich Service Registry-Knoten als Home Agent wählen.

[9] Um Dienste zu finden, die nicht soweit repliziert wurden, dass der benachbarte Service Registry-Knoten sie gespeichert hat, müssen Anfragen im Service Registry-Netz weitergeleitet werden, was die Antwortzeit entsprechend verlängert (s. Kapitel 5.2.4).

Ausschluss von Knoten Wie in den Anforderungen in Kapitel 2.8 beschrieben, kann es in einem drahtlosen Kommunikationssystem im Produktionsbereich Knoten geben, die nicht über eine feste Energieversorgung verfügen. Da ein Knoten, der als Service Registry fungiert, zusätzliche Nachrichten austauschen muss, um die Service Registry aktuell zu halten und Service Registry-Anfragen zu beantworten, benötigen diese Knoten mehr Energie. Der Algorithmus soll es daher erlauben, dass eine Menge von Knoten V_{excl} von der Wahl zum Service Registry-Knoten ausgeschlossen wird. Neben batteriebetriebenen Knoten könnten auch Knoten ausgeschlossen werden, die über zu wenig Speicher verfügen, um die Service Registry zu speichern. Dieser Ausschluss soll per statischer Konfiguration bei der Installation des Netzwerks definiert werden.
Durch den Ausschluss von Knoten kann unter Umständen nicht mehr sichergestellt werden, dass jeder Knoten entweder selbst Service Registry ist oder einen Service Registry-Knoten als direkten Nachbarn hat. Würden beispielsweise alle Knoten ausgeschlossen, wäre diese Forderung offensichtlich unerfüllbar. Da im Produktionsbereich die Topologie des Netzwerks aber nicht rein zufällig ist, sondern zu einem Großteil kontrolliert werden kann, sollte es problemlos möglich sein, ausreichend Knoten zu platzieren, die über die nötigen Voraussetzungen verfügen. Der Algorithmus muss aber erkennen, falls die Forderung nicht erfüllbar ist, damit eine entsprechende Fehlermeldung ausgegeben werden kann.

Definieren von Pflicht-Knoten Es kann auch nötig sein, dass bestimmte Knoten auf jeden Fall Teil der Service Registry sein sollen, beispielsweise weil sie als Gateway zu einem drahtgebundenen Netzwerk dienen. Damit der Algorithmus solche Anforderungen erfüllen kann, ist es nötig, dass Pflicht-Knoten manuell bei der Installation des Netzes konfiguriert werden können. Die Menge dieser Knoten wird als V_{obl} bezeichnet. Die Menge der verbleibenden Knoten, welche weder ausgeschlossen wurden noch Pflicht-Knoten darstellen, werden mit V_{opt} bezeichnet.

5.2.2.2 Voraussetzungen und Annahmen

Es kann vorausgesetzt werden, dass allen Knoten im Netzwerk die Topologie der stationären Knoten aus der automatischen Topologie-Bestimmung (s. Kapitel 4.2.3.2) bekannt ist. Daher ist es nicht nötig, einen verteilten Algorithmus anzuwenden. Vielmehr kann jeder Knoten lokal basierend auf dem globalen Topologiewissen den Algorithmus ausführen und so bestimmen, welche Knoten zur Service Registry gehören und welche Knoten Router sind. Damit alle Knoten zum selben Ergebnis kommen, ist es allerdings wichtig, dass der Algorithmus deterministisch arbeitet. Alternativ kann ein Knoten den Algorithmus ausführen und das Ergebnis verteilen.

Formal wird die Eingabe in den Algorithmus wie folgt definiert:

- V: Menge der Knoten-IDs aller stationärer Knoten
- E: Menge aller symmetrischen (d.h. ungerichteten) Kommunikationslinks, d.h.: $E \subseteq V \times V$ und $\forall (v_1, v_2) \in E : (v_2, v_1) \in E$
- $G = (V, E)$: ungerichteter Kommunikationsgraph aller stationärer Knoten
- V_{obl}: Menge der Knoten-IDs der Knoten, welche Service Registry-Knoten werden müssen
- V_{opt}: Menge der Knoten-IDs der Knoten, welche Service Registry-Knoten werden können
- V_{excl}: Menge der Knoten-IDs der Knoten, welche nicht Service Registry-Knoten werden können

Als Bedingung muss gelten, dass die einzelnen Knoten-Mengen paarweise disjunkt sind und die Vereinigung der drei Mengen die Menge aller Knoten bildet:

$$V_{obl} \cap V_{opt} = \emptyset$$
$$V_{obl} \cap V_{excl} = \emptyset$$
$$V_{opt} \cap V_{excl} = \emptyset$$
$$V_{obl} \cup V_{opt} \cup V_{excl} = V$$

In Abb. 5.1 ist eine beispielhafte Netzwerk-Topologie dargestellt, die eine Eingabe in den Algorithmus darstellen könnte.

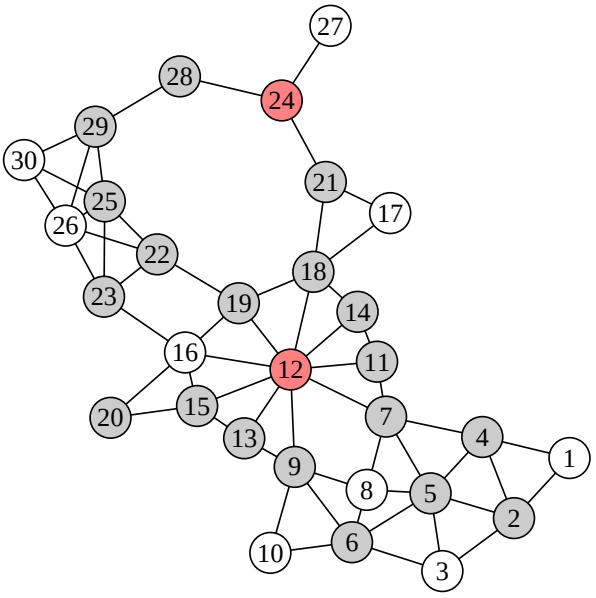


Abbildung 5.1.: Eingabegraph mit Knoten, die Service Registry werden müssen (rot), können (grau) bzw. nicht können (weiß).

5.2.2.3 Das Problem als Clustering-Problem

Der Algorithmus soll unter Beachtung von V_{obl}, V_{opt} und V_{excl} Knoten für die Service Registry auswählen. Alle nicht gewählten Knoten bekommen einen gewählten Nachbarn zugeordnet, über den sie auf die Service Registry zugreifen. Man kann daher sagen, dass um jeden Service Registry-Knoten ein Cluster gebildet wird, welcher alle Knoten enthält, die über diesen Knoten auf die Service Registry zugreifen. In dieser Weise kann das Problem der Knoten-Auswahl als Clustering-Problem betrachtet werden. Unter *Clustering* wird der Prozess verstanden, ein Netzwerk in untereinander verbundene Substrukturen aufzuteilen, welche *Cluster* genannt werden [BBH13]. Die Service Registry-Knoten in V_{SR} koordinieren ihr Cluster und entsprechen daher den *Cluster Heads* (CHs). Alle anderen Knoten in ihrem Cluster, die über sie auf die Service Registry zugreifen, sind ihre *Follower*. Die Knoten in V_{router}, welche die Service Registry-Knoten miteinander verbinden, werden im Clustering-Problem als *Gateway-Knoten* bezeichnet. Obwohl Gateway-Knoten mehrere Cluster miteinander verbinden, gehören auch sie stets zu genau einem Cluster und sind daher auch gleichzeitig Follower. In Abb. 5.2 ist ein Clustering-Beispiel mit zwei Clustern graphisch illustriert.

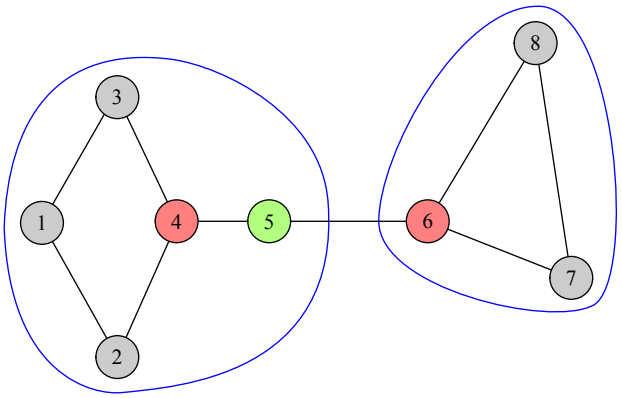


Abbildung 5.2.: Clustering-Beispiel. CHs sind rot, Follower grau und Gateways grün dargestellt. Die blauen Linien umschließen die Mitglieder eines Clusters.

Indem das Problem als Clustering-Problem verstanden wird, kann der gesuchte Algorithmus ebenfalls als Clustering-Algorithmus betrachtet werden. Da das Clustering-Problem gut erforscht ist, gibt es auch eine Reihe von Clustering-Algorithmen (s. [AY07]). Um zu entscheiden, ob es bereits einen geeigneten Algorithmus gibt, werden die in Kapitel 5.2.2.1 gesteckten Ziele zunächst in die Terminologie des Clustering-Problems übersetzt.

Dominating Set Das Ziel, dass jeder Knoten entweder selbst Service Registry-Knoten ist oder einen solchen als direkten Nachbarn hat, entspricht beim Clustering-Problem der Suche nach einem *Dominating Set* [APVH00]. Ein Dominating Set ist eine Teilmenge der Knoten eines Graphen, sodass jeder Knoten des Graphen entweder selbst in der Menge ist oder ein direkter Nachbar eines Knotens in der Menge ist (d.h. es gibt eine Kante zu einem Knoten in der Menge). Bildet ein Clustering-Algorithmus ein Dominating Set, so ist sichergestellt, dass jeder Follower einen direkten Nachbarn hat, der CH ist. Für die verteilte Service Registry sollen also die Knoten, die als Service Registry fungieren, ein Dominating Set bilden, damit jeder Knoten entweder auf seine eigene Service Registry zugreifen kann oder über einen Hop seinen CH erreichen kann, der die Service Registry für ihn bereitstellt.
Im Clustering-Beispiel in Abb. 5.2 bilden die CHs kein Dominating Set des Graphen, da Knoten 1 weder CH noch direkter Nachbar eines CH ist.

Zusammenhängender Graph Es wurde gefordert, dass die Router alle Service Registry-Knoten miteinander verbinden. Dies bedeutet, dass der Graph bestehend aus CHs und Gateways *zusammenhängend* [Vol96] sein muss, d.h. von jedem Knoten in diesem Graphen muss es einen Pfad zu jedem anderen Knoten des Graphens geben. Im Beispiel in Abb. 5.2 ist der Teilgraph bestehend aus den CHs und Gateway-Knoten (Knoten 4, 5 und 6) zusammenhängend.

Three-Hop-Connected Dominating Set In einem Netzwerk, in dem die CHs ein Dominating Set bilden und über Gateways miteinander verbunden sind, werden nie mehr als zwei Gateway-Knoten benötigt, um einen CH mit dem am nächsten gelegenen CH zu verbinden. Dies illustriert Abb. 5.3: Angenommen zwischen den CH 1 und CH 5 wären die drei Gateway-Knoten 2, 3 und 4 nötig, um sie zu verbinden. Da Knoten 3 nicht selbst CH ist, muss einer seiner Nachbarn CH sein, da es sich sonst nicht um ein Dominating Set handeln würde. Wenn es einen solchen CH 6 gibt, dann ist allerdings CH 5 nicht mehr der nächste CH von CH 1 sondern CH 6. Allerdings liegen zwischen CH 1 und CH 6 nur 2 Gateway-Knoten, genauso wie zwischen CH 6 und CH 5.

Zwischen einem CH und dem am nächsten gelegenen CH liegen demnach also maximal zwei Gateways bzw. drei Hops. Somit wird nach einem *Three-Hop-Connected Dominating Set* [DW05] gesucht.

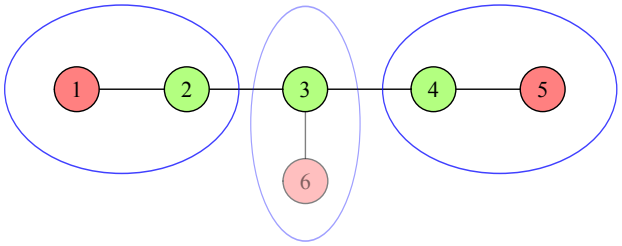


Abbildung 5.3.: Ein über Gateways verbundenes Dominating Set ist stets Three-Hop-Connected

Geringe Cluster-Anzahl Die meisten Clustering-Algorithmen verfolgen in erster Linie das Ziel, möglichst wenige Cluster zu bilden (und damit auch wenige CHs auszuwählen). Auch hier sollen möglichst wenig Service Registry-Knoten ausgewählt werden und damit wenige Cluster gebildet werden.

Es gibt eine ganze Reihe von Clustering-Algorithmen und einige davon verfolgen ähnliche wie die hier genannten Ziele. So gibt es Algorithmen, die ein minimales Dominating Set berechnen, um ein virtuelles Backbone eines Drahtlosnetzwerks zu bilden [DB97, WL99]. Allerdings konnte kein Algorithmus gefunden werden, der es erlaubt, Knoten von der Wahl als CH auszuschließen. Diese Forderung ist aber wichtig, um batteriebetriebene Knoten zu ermöglichen. Es ist auch nicht möglich, diese Knoten einfach aus dem Ausgangsgraphen zu entfernen. Denn fügt man sie zum Ergebnisgraphen wieder hinzu, so ist nicht mehr garantiert, dass die gewählten CHs dann noch ein Dominating Set bilden.

5.2.2.4 Algorithmus

Da kein existierender Algorithmus gefunden werden konnte, der die gesteckten Ziele erfüllt, wurde ein neuer Algorithmus entwickelt. Im Folgenden wird dieser vorgestellt und seine Arbeitsweise an einem Beispiel illustriert.

Grundidee Der Algorithmus baut zunächst ein annähernd minimales Dominating Set auf. Dann fügt er zusätzliche Gateway-Knoten hinzu, um die Cluster zu verbinden und einen zusammenhängenden Graphen aus CHs und Gateways zu erhalten. Schlussendlich fügt er weitere Gateways hinzu, die dazu dienen, lange Pfade zwischen zwei CHs zu verkürzen. Im Folgenden werden die einzelnen Schritte des Algorithmus näher erläutert.

Eingabe-Daten Die Eingabedaten bestehen wie in Kapitel 5.2.2.2 beschrieben aus der Netzwerk-Topologie, die allen Knoten aus der automatischen Topologie-Bestimmung bekannt ist, sowie der Konfiguration, welche Knoten CH bzw. Service Registry werden müssen (V_{obl}), können (V_{opt}) bzw. nicht können (V_{excl}). Abb. 5.1 illustriert graphisch die Eingabedaten des Beispiels.

Ausgabe-Daten Die Ausgabe des Algorithmus umfasst in erster Linie die gewählten CHs ($V_{CH} = V_{SR}$). Außerdem werden Gateways gewählt, welche zusammen mit den CHs als Router bezeichnet werden (V_{router}). Damit klar ist, zu welchem CH ein Follower gehört, wird weiterhin eine

Menge von Links E_{follow} gebildet, welche alle Follower eindeutig mit dem CH verbindet, zu dessen Cluster sie gehören.

Zusammengefasst beinhaltet die Ausgabe folgende Mengen und Graphen:

- V_{CH}: Knoten, welche als CH ausgewählt wurden. V_{CH} ist ein Dominating Set.
  Es gilt $V_{obl} \subseteq V_{CH}$ und $V_{CH} \subseteq V_{obl} \cup V_{opt}$.
- V_{router}: Knoten, welche als Router ausgewählt wurden.
  Es gilt $V_{CH} \subseteq V_{router}$ und $V_{router} \subseteq V_{obl} \cup V_{opt}$.
- E_{router}: ungerichtete Kommunikationslinks, welche Knoten in V_{router} verbinden.
  Es gilt $\forall (v_1, v_2) \in E$ mit $v_1 \in V_{router} \wedge v_2 \in V_{router} : (v_1, v_2) \in E_{router}$.
- $G_{router} = (V_{router}, E_{router})$: ungerichteter Kommunikationsgraph des Router-Netzes (vollständiger Teilgraph[10] von G, der zusammenhängend ist).
- E_{follow}: Menge von ungerichteten Kommunikationslinks, welche alle Follower und alle Gateways mit ihren CHs verbinden. Es gilt $E_{follow} \subseteq E$.
- E_{out}: Menge von ungerichteten Kommunikationslinks, welche die Router miteinander verbinden und die Follower mit ihren CHs verbinden. Es gilt $E_{out} = E_{router} \cup E_{follow}$.
- $G_{out} = (V, E_{out})$: ungerichteter Kommunikationsgraph des Router-Netzes sowie aller Follower.

Schritt 1: Pflicht-Knoten auswählen Im ersten Schritt werden die Pflicht-Knoten in V_{obl} zu CHs gemacht (s. Listing 5.1). Die verwendete Prozedur makeRouter (s. Listing 5.8) fügt dabei nicht nur die Knoten aus V_{obl} den Mengen V_{CH} und V_{router} hinzu, sondern ergänzt den Ausgabegraphen G_{out} auch um die von ihnen abgedeckten Knoten inkl. Verbindungslink und verknüpft die Knoten im Router-Graphen G_{router} mit umliegenden Routern. Darüber hinaus aktualisiert makeRouter die Partitionskennungen im Array P. Unter einer Partition wird ein zusammenhängender Teilgraph von G_{router} verstanden. Jedem Router wird im Array P die Kennung der Partition zugeordnet, zu der der Knoten aktuell gehört. Die Partitionskennung ergibt sich aus der kleinsten Knoten-ID der zur Partition gehörenden Knoten (s. Listing 5.8 und Listing 5.9).

Das Ergebnis des ersten Schrittes im Beispiel ist in Abb. 5.4 visualisiert. Die Pflicht-Knoten 12 und 24 bilden zunächst jeweils ihre eigene Partition. Die Follower-Knoten spielen bei der Partitionierung keine Rolle.

Damit der Algorithmus auf allen Knoten zum selben Ergebnis kommt, muss die ForEach-Schleife über die Elemente der Menge V_{obl} in einer deterministischen Reihenfolge iterieren. Es wird davon ausgegangen, dass die Elemente anhand der Knoten-ID geordnet durchgegangen werden. Das gleiche gilt für alle weiteren ForEach-Schleifen.

```
1 /* Annahme: Knoten in V sind von 0 bis |V|-1 nummeriert. Alternativ kann das
     größte Element aus V zur Bemessung von P genutzt werden. */
2 V_out = ∅; E_out = ∅; V_CH = ∅; V_router = ∅; E_router = ∅; E_follow = ∅; P = intArray[|V|];
3
4 /* Alle Pflicht-Knoten werden als CH ausgewählt und von ihnen abgedeckte Knoten
     zu G_out hinzugefügt */
5 ForEach v in V_obl Do
6     makeRouter(v, true);
7
```

[10] Damit ist hier gemeint, dass zu allen im Teilgraph enthaltenen Knoten auch die Links des Gesamtgraphs enthalten sind, welche diese verbinden. Formal: $\forall (v_1, v_2) \in E$ mit $v_1 \in V_{router} \wedge v_2 \in V_{router} : (v_1, v_2) \in E_{router}$. Es ist also keine Clique gemeint.

```
8 /* G_router enthält jetzt alle Knoten in V_obl. */
9 /* G_out enthält jetzt alle Knoten in V_obl sowie die von ihnen abgedeckten Knoten
      inkl. Verbindungslink. */
```

Listing 5.1: Schritt 1: Pflicht-Knoten auswählen

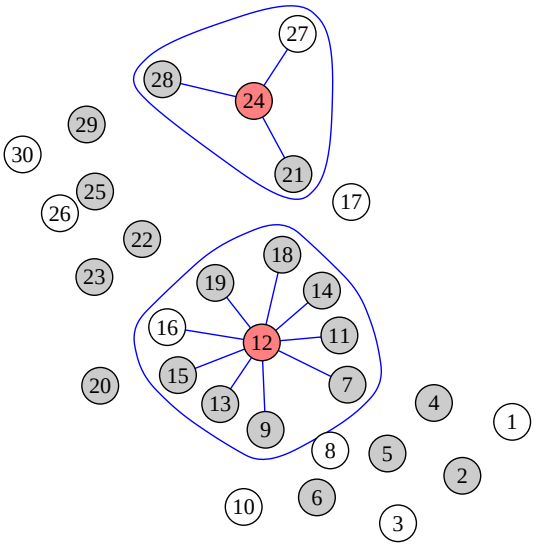


Abbildung 5.4.: Ergebnis von Schritt 1: Die Pflicht-Knoten bilden die ersten Cluster (blau umrahmt).

Schritt 2: Bilden eines Dominating Sets Als nächstes bildet der Algorithmus ein Dominating Set. Dazu macht er solange Kann-Knoten zu CHs, bis der ganze Graph abgedeckt ist (s. Listing 5.2). Zunächst werden dabei jene Knoten als CH gewählt, welche die meisten Follower erhalten. Diese Heuristik hält die Cluster-Anzahl möglichst gering. In Abb. 5.5 ist das Dominating Set im Beispiel illustriert. Wie man sieht, ist der Graph aus CHs noch nicht zwingend zusammenhängend, noch existieren vier Partitionen mit den Kennungen 2, 12, 24 und 25.

```
1  /* Mache optionale Knoten zu CHs bis der ganze Graph abgedeckt ist (d.h. bis V_CH
       ein Dominating Set ist). */
2  While V_out ≠ V Do
3      Begin
4      v_mostNewNodes = undefined;
5      n_mostNewNodes = 0;
6      /* Durchlaufe alle Knoten, die noch CH werden können, es aber noch nicht
           sind */
7      ForEach v in V_opt \ V_CH Do
8          Begin
9          /* Bestimme wie viele neue Knoten mit v abgedeckt werden könnten */
10         n_newNodes = 0;
11         ForEach (v1,v2) in E mit v1 = v und v2 ∉ V_out Do
12             n_newNodes++;
13         /* Muss der Knoten selbst noch abgedeckt werden? */
14         If v ∉ V_out Then
15             n_newNodes++;
16         If n_newNodes > n_mostNewNodes ∨ (n_newNodes = n_mostNewNodes ∧ v ∈ V_out ∧ v_mostNewNodes ∉ V_out) Then
17             Begin
18             /* Bisher deckt v die meisten neuen Knoten ab. Bei Gleichstand
                   werden Knoten bevorzugt, die bereits abgedeckt sind, um zu
                   erreichen, dass G_out weniger stark partitioniert wird. */
```

```
19                  n_mostNewNodes = n_newNodes;
20                  v_mostNewNodes = v;
21                  End
22              End
23          If v_mostNewNodes == undefined Then
24              /* Es wurde kein Knoten gefunden, der neue Knoten abdeckt, aber V_out
                   enthält noch nicht alle Knoten. D.h. es gibt kein Dominating Set aus
                   Knoten in V_obl ∪ V_opt. */
25              Throw NoDominatingSetExists;
26          Else
27              /* Mache den Knoten, der die meisten neuen Knoten abdeckt zum CH */
28              makeRouter(v_mostNewNodes, true);
29          End
30  /* V_CH ist jetzt ein Dominating Set, aber G_router ist nicht zwangsläufig
       zusammenhängend */
31  /* E_follow enthält noch Links zwischen CHs. Entferne diese. */
32  E_follow = E_follow \ E_router;
```

Listing 5.2: Schritt 2: Bilden eines Dominating Sets

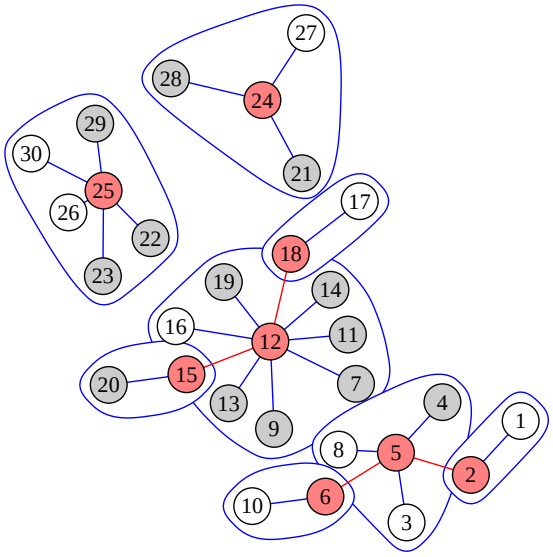


Abbildung 5.5.: Ergebnis von Schritt 2: Ein Dominating Set mit allen CHs (rot) ist gebildet.

Schritt 3: Verknüpfen von Partitionen (1) Als nächstes sollen mehr Partitionen miteinander verbunden werden, indem weitere Follower zu Gateways gemacht werden. Dazu werden zunächst jene Knoten gewählt, welche an mindestens zwei noch nicht verbundene CHs grenzen und diese so verbinden können (s. Listing 5.3). Knoten, die die meisten Partitionen verbinden können, werden zuerst hinzugefügt. Durch diese Heuristik werden die Partitionen mit möglichst wenigen Gateways verknüpft. Das Ergebnis dieses Schritts ist für das betrachtete Beispiel in Abb. 5.6 illustriert. Wie man am Beispiel sieht, sind noch nicht zwingend alle Cluster miteinander verbunden, es gibt noch zwei Partitionen mit den Kennungen 2 und 25.

```
1  /* Suche kann-Knoten, welche noch kein Router sind, aber direkte Nachbarn
       mindestens zweier noch nicht verbundener CHs sind. Von diesen werden zuerst
       jene zu Routern, die die meisten Partitionen verbinden. */
2  moreCandidates = (V_opt \ V_router ≠ ∅);
3  While moreCandidates ∧ notStronglyConnected() Do
```

```
4     Begin
5     /* Es müssen mindestens 2 Partitionen verknüpft werden */
6     n_mostConnectedPartitions = 1;
7     v_mostConnectedPartitions = undefined;
8     ForEach v in V_opt \ V_router Do
9         Begin
10        /* Bestimme die (unverbundenen) Nachbar-Partitionen */
11        P_neighbours = ∅;
12        ForEach (v1,v2) in E mit v1 = v ∧ v2 ∈ V_CH Do
13            P_neighbours = P_neighbours ∪ {P[v2]};
14        If |P_neighbours| > n_mostConnectedPartitions Then
15            Begin
16            /* Der Knoten v verbindet bisher die meisten Partitionen */
17            n_mostConnectedPartitions = |P_neighbours|;
18            v_mostConnectedPartitions = v;
19            End
20        End
21    /* Mache den Knoten zum Router, der die meisten Partitionen verbindet */
22    If v_mostConnectedPartitions ≠ undefined Then
23        makeRouter(v_mostConnectedPartitions, false);
24    Else
25        moreCandidates = false;
26    End
27 /* Es wurden jetzt alle Partitionen verbunden, welche durch Hinzufügen nur eines
      Routers verbunden werden können. */
```

Listing 5.3: Schritt 3: Verknüpfen von Partitionen (1)

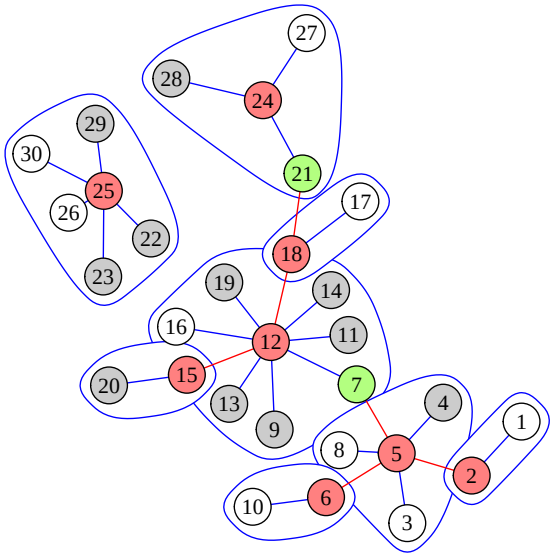


Abbildung 5.6.: Ergebnis von Schritt 3: Knoten, welche mindestens zwei Partitionen verbinden, werden zu Gateways (grün).

Schritt 4: Verknüpfen von Partitionen (2) Um alle CHs über Gateways zu verbinden, werden als nächstes Paare von Followern als Gateways ausgewählt (s. Listing 5.4). Wie in Abb. 5.3 illustriert wurde, können auf diese Weise alle CHs über Gateways verbunden werden. Das Ergebnis dieses Schritts im Beispiel ist in Abb. 5.7 zu sehen. Das gesamte Router-Netz bildet nun nur noch eine Partition mit der Kennung 2.

```
1  /* Suche Paare von noch nicht gewählten kann-Knoten, die (mindestens) zwei CHs
       unterschiedlicher Partitionen verbinden. */
2  /* Durchlaufe alle noch nicht gewählten kann-Knoten */
3  ForEach v in V_opt \ V_router Do
4      Begin
5      /* Suche benachbarte noch nicht gewählte kann-Knoten */
6      ForEach (v_1,v_2) in E mit v_1 = v ∧ v_2 ∈ (V_opt \ V_router) Do
7          Begin
8          /* v_1 und v_2 sind kann-Knoten, die noch nicht gewählt sind. Prüfe, ob sie
               an zwei unterschiedliche Partitionen grenzen. */
9          ForEach (v_3,v_4) in E mit v_3 = v_1 ∧ v_4 ∈ V_CH Do
10             Begin
11             /* v_1 grenzt an den CH v_4. */
12             ForEach (v_5,v_6) in E mit v_5 = v_2 ∧ v_6 ∈ V_CH Do
13                 Begin
14                 /* v_2 grenzt an den CH v_6. */
15                 If P[v_4] ≠ P[v_6] Then
16                     Begin
17                     /* v_4 und v_6 gehören zu unterschiedlichen Partitionen, also
                           mache v_1 und v_2 zu Routern, um sie zu verbinden. */
18                     makeRouter(v_1, false);
19                     makeRouter(v_2, false);
20                     End
21                 End
22             End
23         End
24     End
25 /* Prüfe ob G_router tatsächlich zusammenhängend ist. Wenn nicht existiert kein
       zusammenhängender Graph G_router mit Knoten aus V_obl ∪ V_opt. */
26 If notStronglyConnected() Then
27     Throw NotStronglyConnected;
28 /*Nun ist G_router ein zusammenhängender Graph und V_CH ein Dominating Set.*/
```

Listing 5.4: Schritt 4: Verknüpfen von Partitionen (2)

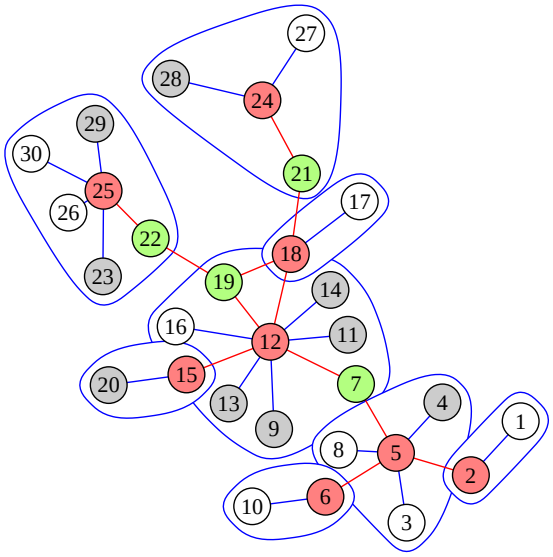


Abbildung 5.7.: Ergebnis von Schritt 4: Paare von Knoten, welche mindestens zwei bisher unverbundene Partitionen verbinden, werden zu Gateways (Knoten 19 und 22).

Schritt 5: Verkürze Pfade zwischen CHs (1) Alle CHs sind bereits über Gateways verknüpft. Zwischen einigen CHs sind die Pfade über Router allerdings noch unnötig lang. Daher sollen zusätzliche Gateways hinzugefügt werden, falls diese einen Pfad zwischen zwei CHs verkürzen. In diesem Schritt werden solche Knoten als Gateways ausgewählt, die direkt zwei CHs miteinander verbinden können, die bisher über längere Pfade verbunden sind (s. Listing 5.5). Abb. 5.8 zeigt das Ergebnis dieses Schrittes im Beispiel.

```
1  /* Es werden noch zusätzliche Router hinzugefügt, um Pfade im Router-Netzwerk zu
       verkürzen */
2  /* Zunächst werden Knoten hinzugefügt, die zwei noch nicht direkt oder über
       einen Gateway-Knoten miteinander verbundene CHs verbinden */
3
4  /* Durchlaufe alle noch nicht gewählten kann-Knoten */
5  ForEach v in V_opt \ V_router Do
6      Begin
7      /* Suche benachbarte CHs */
8      ForEach (v_1,v_2) in E mit v_1 = v ∧ v_2 ∈ V_CH Do
9          Begin
10         /* v grenzt an den CH v_2 */
11         ForEach (v_3,v_4) in E mit v_3 = v ∧ v_4 ≠ v_2 ∧ v_4 ∈ V_CH Do
12             Begin
13             /* v grenzt an die CHs v_2 und v_4 */
14             /* Prüfe, ob v_2 und v_4 bereits direkt oder über einen weiteren Knoten
                   v_6 verbunden sind, und mache andernfalls v zum Router */
15             If ∄(v_5,v_6) ∈ E_router : (v_5 = v_2 ∧ ((v_6 = v_4) ∨ (v_6,v_4) ∈ E_router)) Then
16                 makeRouter(v,false);
17             End
18         End
19     End
```

Listing 5.5: Schritt 5: Verkürze Pfade zwischen CHs (1)

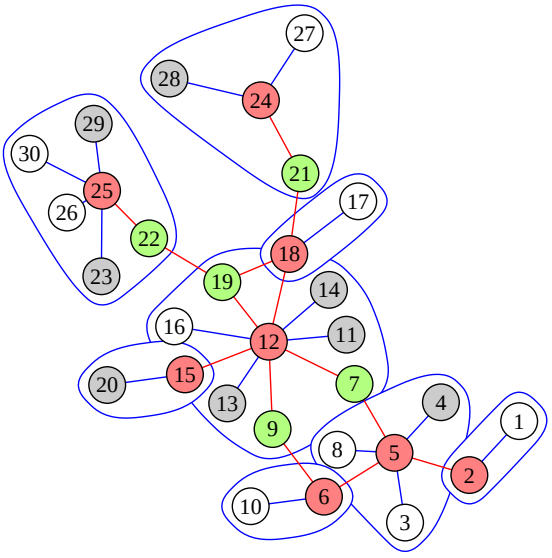


Abbildung 5.8.: Ergebnis von Schritt 5: Weitere Follower, die direkt zwei Cluster verbinden, werden zu Gateways, um Pfade zwischen CHs zu verkürzen (Knoten 9).

Schritt 6: Verkürze Pfade zwischen CHs (2) Im letzten Schritt werden Paare von Knoten hinzugefügt, welche zwei CHs verbinden können, die bisher nur über längere Pfade verbunden sind (s. Listing 5.6). Das Ergebnis zeigt Abb. 5.9 anhand des Beispiels.

```
1  /* Füge Paare von Knoten hinzu, welche zwei CHs verbinden, die noch nicht direkt
       oder über einen Router verbunden sind */
2  /* Durchlaufe alle noch nicht gewählten kann-Knoten */
3  ForEach v in V_opt \ V_router Do
4      Begin
5      /* Suche benachbarten nicht gewählen Kann-Knoten */
6      ForEach (v_2,v_3) in E mit v_2 = v ∧ v_3 ∈ V_opt \ V_router Do
7          Begin
8          /* v grenzt an den Kann-Knoten v_3, der noch kein Router ist */
9          ForEach (v_4,v_5) in E mit v_4 = v ∧ v_5 ∈ V_CH Do
10             Begin
11             /* v grenzt an den CH v_5 */
12             ForEach (v_6,v_7) in E mit v_6 = v_3 ∧ v_7 ∈ V_CH ∧ v_7 ≠ v_5 do
13                 Begin
14                 /* v_3 grenzt an einen anderen CH v_7 */
15                 /* Prüfe, ob v_5 und v_7 bereits direkt oder über einen oder zwei
                       Router verbunden sind */
16                 If ∄(v_8,v_9) ∈ E_router : v_8 = v_5 ∧ (
17                         (v_9 = v_7)∨                                  /* v_5 ↔ v_7 */
18                         (v_9,v_7) ∈ E_router∨                         /* v_5 ↔ v_9 ↔ v_7 */
19                         (∃(v_10,v_11) ∈ E_router : (v_9,v_10) ∈ E_router ∧ v_11 = v_7)  /* v_5 ↔ v_9 ↔ v_10 ↔ v_7 */
20                     ) Then
21                     Begin
22                     makeRouter(v,false);
23                     makeRouter(v_3,false);
24                     End
25                 End
26             End
27         End
28     End
```

Listing 5.6: Schritt 6: Verkürze Pfade zwischen CHs (2)

Hilfsfunktionen: notStronglyConnected Diese in Listing 5.7 aufgeführte Funktion prüft, ob der Graph G_{router} zusammenhängend ist. Dazu werden alle Router durchlaufen und es wird überprüft, ob sie derselben Partition angehören. Wird ein Router gefunden, der einer anderen Partition angehört als die bisher betrachteten, so ist der Graph nicht zusammenhängend.

```
1  Boolean Function notStronglyConnected()
2      Begin
3      p = undefined;
4      ForEach v in V_router Do
5          Begin
6          If p == undefined Then
7              p = P[v];
8          ElseIf p ≠ P[v] Then
9              /* Mindestens 2 Knoten gehören nicht zur selben Partition, also ist
                   G_router nicht strongly connected. */
10             Return true;
11         End
12     Return false;
13     End
```

Listing 5.7: Hilfsfunktion notStronglyConnected

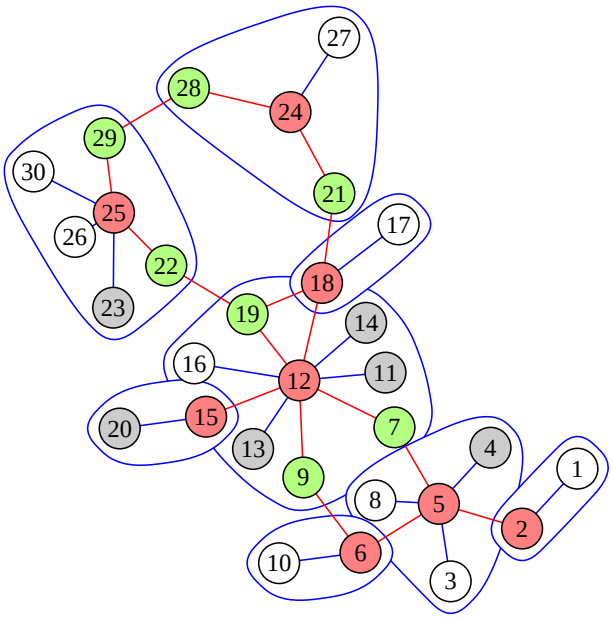


Abbildung 5.9.: Ergebnis von Schritt 6: Paare von Followern, welche zwei Cluster verbinden, werden zu Gateways, um Pfade zwischen CHs zu verkürzen (Knoten 28, 29).

Hilfsprozedur: makeRouter Diese in Listing 5.8 gezeigte Hilfsprozedur macht den Knoten v zu einem Router und optional auch zu einem CH. Falls der Knoten CH wird, werden die von ihm abgedeckten Nachbarknoten, die seine Follower werden, inkl. Verbindungslink zu G_{out} hinzugefügt. Außerdem wird der neue Router mit den Nachbar-Routern verknüpft (E_{router}) sowie die Partitions-Kennung aktualisiert.

```
1  Procedure makeRouter(int v, bool CH)
2      Begin
3      V_router = V_router ∪ {v};
4      V_out = V_out ∪ {v};
5      If CH Then
6          V_CH = V_CH ∪ {v};
7      /* Wenn v an keinen anderen Router angrenzt, bildet er seine eigene
           Partition */
8      P[v] = v;
9
10     /* Durchlaufe alle Kanten, die von v ausgehen */
11     ForEach (v1,v2) in E mit v1 = v Do
12         Begin
13         /* Füge noch nicht abgedeckte Knoten zu G_out hinzu falls v CH werden soll
               */
14         If CH ∧ v2 ∉ V_out Then
15             Begin
16             V_out = V_out ∪ {v2};
17             E_out = E_out ∪ {(v1,v2),(v2,v1)};
18             E_follow = E_follow ∪ {(v1,v2),(v2,v1)};
19             End
20         /* Verknüpfe den Router mit direkten Nachbar-Routern in G_router. */
21         If v2 ∈ V_router Then
22             Begin
23             E_router = E_router ∪ {(v1,v2),(v2,v1)};
24             E_out = E_out ∪ {(v1,v2),(v2,v1)};
25             If P[v2] < P[v] Then
```

```
26                  /* Der Knoten grenzt an einen Router mit einer kleineren
                        Partitions-Kennung -> übernehme diese */
27                  P[v] = P[v_2];
28              End
29          End
30      updatePartition(v,P[v]);
31      End
```

Listing 5.8: Hifsprozedur makeRouter

Hilfsfunktionen: updatePartition Ordnet dem Knoten v und allen Routern, welche mit v verbunden sind (d.h. zur selben Partition gehören), die Partitionskennung p zu (s. Listing 5.9). Die kleinste Partitionskennung setzt sich immer durch, d.h. es werden immer nur die Knoten aktualisiert, die noch eine größere Kennung haben.

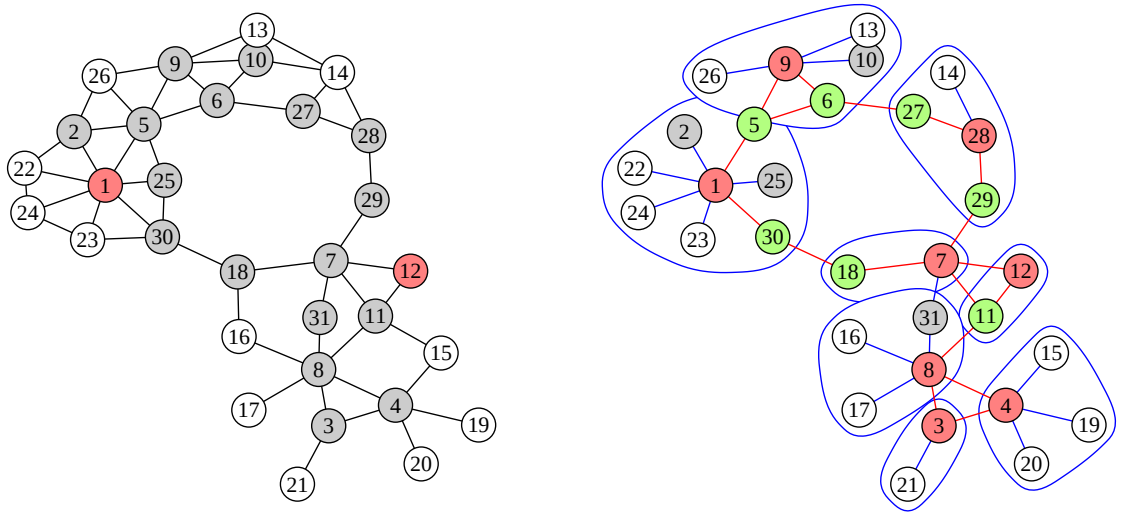
```
1 Procedure updatePartition(v,p)
2     Begin
3     P[v] = p;
4     ForEach (v_1,v_2) in E_router mit v_1 = v und P[v_2] > p Do
5         updatePartition(v_2,p);
6     End
```

Listing 5.9: Hilfsprozedur updatePartition

5.2.2.5 Evaluation

Das Ergebnis des Algorithmus im gezeigten Beispiel (s. Abb. 5.9) zeigt anschaulich, dass der Algorithmus die gesteckten Ziele erreicht: Es wird ein Three-Hop-Connected Dominating Set gebildet, d.h. jeder Knoten ist CH oder hat einen CH als Nachbarn und alle CHs sind miteinander über maximal zwei Gateways miteinander verbunden. Es werden keine ausgeschlossenen Knoten aus V_{excl} zu CHs oder Gateways gewählt und alle Pflicht-Knoten aus V_{obl} werden als CHs ausgewählt. Abb. 5.10 zeigt den Eingabe- und Ausgabegraphen eines weiteren Beispiels, bei dem die genannten Ziele ebenfalls erreicht werden.

In beiden Beispielen wurden acht CHs gewählt, von denen zwei bereits als Pflicht-CH konfiguriert waren. Von den 30 bzw. 31 Knoten in den Beispielen wurden also wie gefordert recht wenige Knoten als CH ausgewählt. Es ist nicht Ziel dieses Algorithmus, immer die minimal mögliche Anzahl CHs auszuwählen. Anhand eines Beispiels lässt sich zeigen, dass der Algorithmus dies auch nicht in jedem Fall erreicht. Den Eingabegraphen dieses Beispiels zeigt Abb. 5.11. Ein minimales Dominating Set besteht in diesem Fall aus zwei CHs, nämlich den Knoten 2 und 5, wie es Abb. 5.12 illustriert. Der Algorithmus bestimmt in Schritt 2 beim Bilden eines Dominating Sets für jeden Knoten, wie viele bisher unabgedeckte Knoten durch diesen abgedeckt würden. Zunächst betrachtet er Knoten 1, der 2 Nachbarknoten und sich selbst, also 3 Knoten abdecken würde. Gleiches gilt für die Knoten 2, 4 und 5. Die Knoten 3 und 6 würden nur einen Nachbarn und sich selbst, also 2 Knoten abdecken. Da Gleichstand zwischen den Knoten 1, 2, 4 und 5 besteht und bisher noch keine anderen Knoten gewählt wurden, wählt der Algorithmus den zuerst betrachteten Knoten 1 aus. Diese Wahl führt allerdings dazu, dass anstatt eines weiteren CH zwei weitere CHs gewählt werden müssen, um ein Dominating Set zu bilden, wie es in Abb. 5.13 zu sehen ist. Ob ein minimales Ergebnis entsteht, hängt in diesem Beispiel von der Reihenfolge ab, in welcher der Algorithmus die Knoten prüft, und damit von der Nummerierung der Knoten.



(a) Eingabegraph (b) Ausgabegraph

Abbildung 5.10.: Ein- und Ausgabegraphen eines weiteren Beispiels

Abbildung 5.11.: Eingabegraph eines Beispiels, bei dem kein minimales Dominating Set gebildet wird

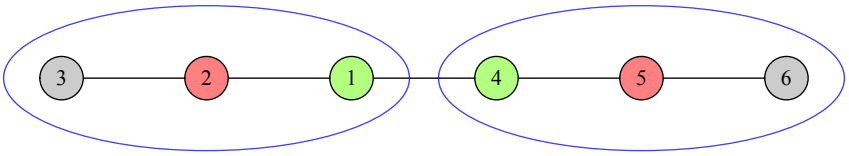


Abbildung 5.12.: Minimales Dominating Set zum Eingabegraphen in Abb. 5.11

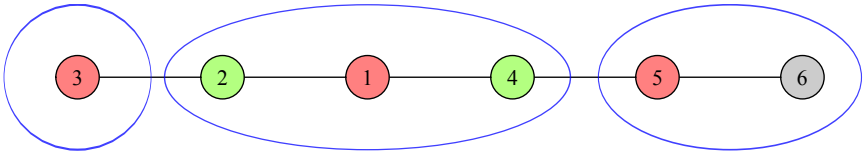


Abbildung 5.13.: Nicht minimales Ergebnis des Algorithmus zum Eingabegraphen in Abb. 5.11

Der Algorithmus verwendet zwar Heuristiken, um möglichst wenige Knoten als CH auszuwählen, allerdings stellen diese, wie das Beispiel zeigt, nicht in jedem Fall sicher, dass ein minimales Dominating Set gebildet wird. Für die Anwendung in einem drahtlosen Kommunikationssystem im Produktionsbereich ist es auch nicht notwendig, dass in jedem Fall die minimale Anzahl CHs gewählt wird. Zwar führt eine größere Anzahl CHs bzw. Service Registry-Knoten dazu, dass mehr Nachrichten ausgetauscht werden müssen, um die Service Registry aktuell zu halten. Solange es sich aber nur um wenige zusätzliche Knoten handelt, fällt dies wenig ins Gewicht. Wichtiger ist dagegen, dass Knoten, die nicht über eine feste Energieversorgung verfügen, nicht mehr als nötig belastet werden. Durch zusätzliche Service Registry-Knoten werden diese Knoten, die stets Follower werden und daher nicht am Nachrichtenaustausch innerhalb der Registry beteiligt sind, aber nicht stärker belastet.

In den Schritten 5 und 6 versucht der Algorithmus, Pfade zwischen den CHs zu verkürzen, indem weitere Gateways hinzugefügt werden. Das führt zu einer schnelleren Ausbreitung von Information und zu einem stärker zusammenhängenden Netzwerk. Dies ist insbesondere dann relevant, wenn Dienstinformationen wie in Kapitel 5.2.1.1 genannt nur über n Hops auf dem Service Registry-Netzwerk repliziert werden sollen. Im ersten Beispiel sieht man, dass in Abb. 5.9 zwischen den CHs 25 und 24 der Pfad mit drei Hops deutlich kürzer ist als zuvor in Abb. 5.8 mit fünf Hops. Im zweiten Beispiel in Abb. 5.10 wurden u.a. die zusätzlichen Gateways 18 und 30 hinzugefügt, wodurch der Pfad von CH 1 zu CH 7 von sechs auf drei Hops verkürzt werden konnte. Würden Dienste nur über $n = 3$ Hops verbreitet, würde beispielsweise im ersten Beispiel ein Dienst von CH 25 nicht zu CH 24 verbreitet werden, sondern müsste über eine aufwendigere Suche über alle Service Registry-Knoten gesucht werden (s. Kapitel 5.2.4).

Anstatt Gateways zu nutzen, wäre es auch denkbar, alle vom Algorithmus als Router gewählten Knoten als CHs zu wählen, also auch die Gateway-Knoten. Dies würde dazu führen, dass anstatt eines Three-Hop-Connected Dominating Sets ein One-Hop-Connected Dominating Set gebildet würde. Es ist allerdings nicht nötig, dass zusätzliche Knoten Service Registry-Informationen speichern, da durch das Dominating Set bereits sichergestellt ist, dass jeder Knoten einen direkten Nachbarn hat, der Teil des Service Registry-Netzwerks ist. Als Gateways sind diese Knoten allerdings nötig, um Informationen zwischen den Service Registry-Knoten austauschen zu können.

5.2.2.6 Terminierung

Der Algorithmus terminiert unabhängig von den Eingabedaten, auch in Fällen, in denen kein Three-Hop-Connected Dominating Set gebildet werden kann. Es können drei Fälle unterschieden werden, wie der Algorithmus terminieren kann:

1. Es wurde erfolgreich ein Three-Hop-Connected Dominating Set gebildet.
2. Es kann kein Dominating Set gebildet werden.
3. Es kann kein Three-Hop-Connected Dominating Set gebildet werden.

Im ersten Fall durchläuft der Algorithmus erfolgreich alle 6 Schritte und kommt zum gewünschten Ergebnis. Dieser Fall wurde in Kapitel 5.2.2.4 im Detail beschrieben.

Im zweiten Fall erkennt der Algorithmus in Schritt 2 (s. Listing 5.2, Zeilen 23-25), dass es keinen Knoten in $V_{opt} \setminus V_{CH}$ gibt, der einen weiteren Knoten abdeckt, aber trotzdem $V_{out} \neq V$ ist, d.h. es gibt Knoten, die weder selbst CH werden können noch Nachbarknoten haben, die CH werden können. Dieser Fall kann eintreten, wenn ein oder mehrere Knoten aus V_{excl} als direkte Nachbarn nur andere Knoten aus V_{excl} haben und daher für sie kein CH gefunden werden kann. Abb. 5.14

illustriert ein solches Beispiel: Für Knoten 1 kann hier kein CH gefunden werden, da der einzige benachbarte Knoten 2 ebenfalls von der Wahl als CH ausgeschlossen wurde. Der Algorithmus bricht in diesem Fall in Schritt 2 mit dem Fehler *NoDominatingSetExists* ab. Als Folge würde das System nicht starten, sondern eine Fehlermeldung anzeigen. Zur Behebung des Problems könnten beispielsweise zusätzliche Knoten platziert werden, die als CH dienen können.

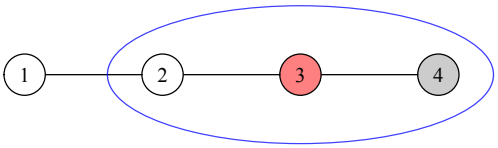


Abbildung 5.14.: Beispiel, in dem kein Dominating Set gebildet werden kann

Im dritten Fall scheitert der Algorithmus in Schritt 4 (s. Listing 5.4) daran, alle CHs direkt oder über Gateways miteinander zu verbinden. Im vorherigen Schritt wurden alle einzelnen noch nicht gewählten Kann-Knoten hinzugefügt, welche mindestens zwei Partitionen verknüpfen können. In diesem Schritt werden alle Paare von noch nicht gewählten Kann-Knoten hinzugefügt, die mindestens zwei Partitionen verbinden können. Wenn der Router-Graph G_{router} auch nach diesem Schritt noch nicht zusammenhängend ist, d.h. es noch mehr als eine Partition gibt, so lässt sich aus diesen Eingangsdaten kein Three-Hop-Connected Dominating Set bilden. Dies ist dann der Fall, wenn entweder der Eingangsgraph schon nicht zusammenhängend ist oder aber zwei Partitionen nur über einen Knoten verbunden werden könnten, der in V_{excl} enthalten und damit von der Wahl als Gateway ausgeschlossen ist. In Abb. 5.15 sind beide Fälle illustriert: Zwischen CH 5 und den anderen CHs kann keine Verbindung geschaffen werden, da der Eingabegraph nicht zusammenhängend ist. Darüber hinaus kann CH 1 nicht über ein Gateway mit CH 3 verbunden werden, da Knoten 2 von der Wahl zum Gateway ausgeschlossen wurde. Wenn das Bilden eines Three-Hop-Connected Dominating Sets nicht möglich ist, bricht der Algorithmus in Schritt 4 mit der Fehlermeldung *NotStronglyConnected* ab (s. Listing 5.4, Zeile 27). Auch in diesem Fall wird beim Start des Systems eine Fehlermeldung angezeigt und es müssen beispielsweise weitere Kann-Knoten hinzugefügt werden.

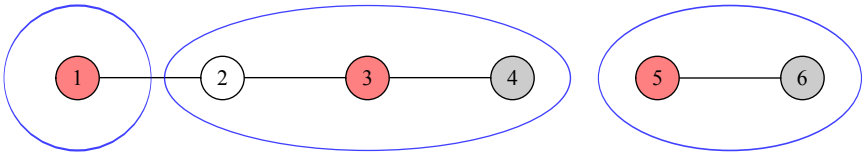


Abbildung 5.15.: Beispiel, in dem die CH eines Dominating Sets nicht über Gateways verbunden werden können

5.2.3 Dienst-Bekanntmachung und Replikation

Da die Service Registry lokal repliziert werden soll (s. Kapitel 5.2.1.1), muss ein Service Registry-Knoten einen neuen Dienst, der ihm bekannt gemacht wird (Publish), mit den umliegenden Service Registry-Knoten austauschen. Ist ein Knoten selbst ein Service Registry-Knoten, so speichert er die Informationen über den neuen Dienst zunächst nur in seiner eigenen Service Registry und vermerkt ihn als neuen Dienst, der noch nicht repliziert wurde. Ist der Knoten kein Service Registry-Knoten, so sendet er eine *Publish-Nachricht* an seinen CH. Da die Menge der CHs ein Dominating Set ist, beträgt die Entfernung eines Knotens zu seinem CH nie mehr als einen Hop. Die Nachricht muss also nicht über andere Knoten weitergeleitet werden. Empfängt ein CH eine

Publish-Nachricht von einem seiner Follower, so speichert er die in der Nachricht enthaltenen Dienstinformationen in seiner eigenen Service Registry und vermerkt sie als neu.

Die lokale Replikation findet nun über eine *regelmäßige Synchronisation der Registry* statt. In regelmäßigen, konfigurierbaren Abständen, z.B. alle 2 Sekunden, sendet ein Service Registry-Knoten eine Publish-Nachricht per lokalen Broadcasts an alle Nachbar-Router, d.h. CHs und Gateway-Knoten. In dieser Nachricht sind zunächst jene Dienste enthalten, die als neu vermerkt wurden und noch nicht repliziert wurden. Ist in der Nachricht noch Platz für weitere Dienste, füllt der Knoten die Nachricht mit bereits zuvor replizierten Diensten aus seiner Registry auf. Er iteriert dabei durch die Dienste in seiner Registry und fährt beim nächsten Synchronisationsvorgang an der Stelle fort, an der er beim letzten Mal aufgehört hat, sodass sichergestellt wird, dass alle Dienste der Reihe nach regelmäßig synchronisiert werden. Dienste, welche n Hops von ihm entfernt liegen und daher nicht weiter repliziert werden sollen, überspringt er dabei. Empfängt ein Gateway-Knoten eine solche Publish-Nachricht, inkrementiert er den Hop-Counter der darin enthaltenen Dienste und leitet die Nachricht per lokalem Broadcast weiter. Erreicht oder übersteigt der Hop-Counter aller enthaltenen Dienste den Parameter n, über den Dienste repliziert werden, so verwirft der Gateway-Knoten die Nachricht[11]. Trifft eine solche Publish-Nachricht schließlich bei einem CH ein, so aktualisiert dieser seine Registry anhand der enthaltenen Dienstinformationen. Dabei inkrementiert er zunächst den Hop-Counter für jeden Dienst und prüft, ob der Dienst in n-Hop-Reichweite ist. Ist dies der Fall, wird anhand einer vom Service Provider vergebenen Sequenznummer überprüft, ob bereits aktuellere Informationen über diesen Dienst gespeichert sind. Ist dies nicht der Fall, übernimmt der Knoten die Dienstinformation aus der empfangenen Nachricht in seine Registry und markiert sie als neu, sodass sie beim nächsten Synchronisationsvorgang bevorzugt weitergeleitet wird.

Damit zwei Gateways Nachrichten nicht in einer Schleife austauschen, ist jede Nachricht mit einer vom Sendeknoten[12] vergebenen Sequenznummer versehen. Ein Router merkt sich zu jedem Sendeknoten die bereits weitergeleiteten Sequenznummern in einem Ringspeicher und leitet nur jene Nachrichten weiter, die nicht bereits in diesem Speicher vermerkt sind.

5.2.4 Aufsuchen eines Dienstes

Möchte ein Knoten einen Dienst in der Service Registry nachschlagen und ist selbst kein Service Registry-Knoten, so sendet er eine *Search-Nachricht* an seinen CH. Ist ein Knoten selbst Service Registry-Knoten und möchte einen Dienst suchen oder empfängt eine Search-Nachricht von einem anderen Knoten, so schlägt er den gesuchten Dienst zunächst in seiner eigenen Service Registry nach. Da Dienste nur über n Hops repliziert werden, kann ein Service Registry-Knoten nicht wissen, ob ein Dienst, den er selbst nicht gespeichert hat, von einem weiter entfernten Knoten angeboten wird. Aus diesem Grund muss er eine Search-Nachricht an die anderen Service Registry-Knoten senden. Dazu sendet er sie per lokalem Broadcast an die benachbarten Gateways und CHs. Wird sie von einem Gateway-Knoten empfangen, so leitet dieser die Nachricht per lokalem Broadcast weiter. Wird sie von einem CH empfangen, schlägt dieser den Dienst zunächst in seiner eigenen Service Registry nach und leitet sie ggf. weiter, falls er den gesuchten Dienst nicht gespeichert hat.

[11] Da Gateway-Knoten Publish-Nachrichten nur ganz weiterleiten oder verwerfen, kann es sein, dass der Hop-Counter einzelner Dienste in einer Nachricht den Parameter n übersteigt, während andere enthaltene Dienste ihn noch nicht erreicht haben.

[12] Der Sendeknoten ist der CH, der die Publish-Nachricht ursprünglich versendet hat, nicht ein dazwischen liegendes Gateway.

Da es im Router-Graphen Kreise geben kann (vgl. Abb. 5.9), können Search-Nachrichten einen Knoten auf mehreren Pfaden erreichen. Damit sie nicht mehrfach bearbeitet werden, speichert jeder Router in einem Ringpuffer eine Liste mit Identifiern der bereits bearbeiteten Search-Nachrichten. Trifft eine in dieser Liste gespeicherte Search-Nachricht erneut ein, wird sie verworfen.

Damit Search-Nachrichten nicht unnötig lange weitergeleitet werden müssen, berechnet der CH, der eine Suche beginnt, die maximale Hop-Anzahl, über die eine Search-Nachricht übertragen werden muss, nach Gl. (5.1).

$$hopsMax = max(min(maxCHDist, maxCHDist - n + 2), 0) \tag{5.1}$$

Wie sich diese Gleichung ergibt, wird am in Abb. 5.16 illustrierten Beispiel gezeigt. Gehen wir zunächst davon aus, dass CH 1 eine Suche gestartet hat und nach einem Dienst von Knoten 8 sucht. Knoten 8 hat seinen Dienst bei CH 6 publiziert. Es wird angenommen, dass Dienstinformationen über $n = 2$ Hops um den ursprünglichen CH repliziert werden (s. Kapitel 5.2.1.1), d.h. die Information über den Dienst von Knoten 8 wurde von CH 6 zu CH 5 und CH 4 repliziert. CH 1 berechnet die Entfernung in Hops zum am weitesten entfernten CH 6 ($maxCHDist = 5$) mit Hilfe der jedem Knoten bekannten Topologie-Information. Eine von CH 1 gestartete Suche muss folglich nur bis zu CH 4 weitergeleitet werden, da sämtliche Dienstinformationen von CH 5 und CH 6 zu ihm repliziert werden. So ergibt sich, dass eine Suche von CH 1 über $maxCHDist - n = 5 - 2 = 3$ Hops weitergeleitet werden muss. Der folgende Fall zeigt allerdings, dass dies nicht immer ausreicht.

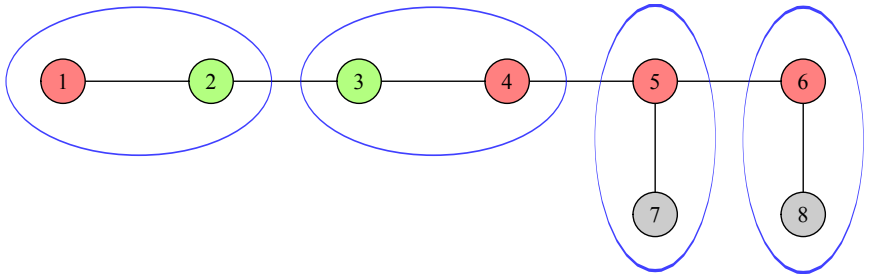


Abbildung 5.16.: Beispiel-Topologie zur Illustration von Gl. (5.1)

Gehen wir nun davon aus, dass CH 6 eine Suche startet und einen Dienst sucht, der von Gateway 2 angeboten wird. Gateway 2 hat seinen Dienst bei CH 1 publiziert. Von dort wird der Dienst über $n = 2$ Hops repliziert, allerdings befinden sich keine CHs im Umkreis von 2 Hops um CH 1. Daher wird die Dienstinformation nicht repliziert. Damit eine Suche von CH 6 Dienste finden kann, die bei CH 1 registriert wurden, ist es also nötig, dass seine Suchanfrage über $maxCHDist - n + 2 = 5 - 2 + 2 = 5$ Hops weitergeleitet wird. Die Konstante 2 ergibt sich daraus, dass in einem Three-Hop-Connected Dominating Set maximal zwei Gateways zwischen zwei CHs möglich sind (s. Kapitel 5.2.2.3). Damit in jedem Fall alle Dienste gefunden werden, erfolgt die Weiterleitung immer über $maxCHDist - n + 2$ Hops, auch wenn dies – wie im obigen Fall – in manchen Fällen weiter als nötig ist.

Ist der Parameter der lokalen Replikation $n < 2$, beispielsweise $n = 1$, so würde sich allerdings mit $maxCHDist - n + 2 = maxCHDist - 1 + 2 = maxCHDist + 1$ ergeben. Es macht aber in keinem Fall Sinn, eine Suchanfrage über mehr als $maxCHDist$ Hops weiterzuleiten. Daher wird das Minimum aus beiden Werten gebildet, um $hopsMax$ zu bestimmen (s. Gl. (5.1)).

Falls n so groß konfiguriert wird, dass es $maxCHDist + 2$ übersteigt, so ist die Registry vollständig repliziert und keine Weiterleitung einer Search-Nachricht nötig. Das Maximum mit 0 sorgt in diesem Fall dafür, dass $hopsMax$ nicht negativ wird.

Bevor ein Router eine Search-Nachricht weiterleitet, prüft er, ob seine Entfernung zu dem CH, der die Suche initiiert hat, bereits $hopsMax$ beträgt. Ist dies der Fall, wird die Nachricht nicht weitergeleitet. Die Entfernung kann er anhand der lokal vorhandenen Topologie bestimmen.

Findet ein Knoten einen gesuchten Dienst in seiner Service Registry, sendet er eine *SearchResult-Nachricht* an den ursprünglich suchenden Knoten. Dies kann ein CH, aber auch ein Follower sein. Die Route, über die diese Nachricht gesendet wird, bestimmt der Network Layer. Empfängt ein Router eine SearchResult-Nachricht, so bearbeitet er ggf. später eintreffende Search- oder SearchResult-Nachrichten zur selben Suchanfrage nicht mehr, da bereits ein Ergebnis für diese Suche gefunden wurde.

Trifft die SearchResult-Nachricht am ursprünglich suchenden Knoten ein, so wird dort das Event ausgelöst, welches mit der Suchanfrage verknüpft war, sodass die Anwendung das Suchergebnis verarbeiten und den gefundenen Dienst ggf. abonnieren oder aufrufen kann. Trifft kein Suchergebnis ein, wird ein Timeout ausgelöst, der ebenfalls das mit der Suchanfrage verknüpfte Event auslöst, um die Anwendung über die fehlgeschlagene Suche zu informieren.

5.3 Implementierung

Der in Kapitel 5.2.2 beschriebene Clustering-Algorithmus sowie die partiell verteilte Service Registry wurden im Rahmen dieser Arbeit in C++ implementiert. Zum Testen des Clustering-Algorithmus wurde dieser um eine Debug-Ausgabe erweitert, welche sich über das igraph-Paket für R [igr] als Graph plotten lässt. Die Abbildungen in Kapitel 5.2.2 wurden auf diese Weise erstellt.

Die verteilte Service Registry baut auf einem im Rahmen einer Masterarbeit [Eng13] entwickelten *BiPS-Framework* für die in Kapitel 4.1 vorgestellte Imote 2-Plattform auf. Da zum Testen der verteilten Service Registry eine Topologie mit mindestens 20 Knoten sinnvoll ist, und nicht so viele Imote 2-Knoten zur Verfügung standen, wird eine BiPS-Integration für den Netzwerksimulator ns-3 [ns-] genutzt. Diese bildet die BiPS-Schnittstellen nach, sodass die Middleware im Simulator die gleichen Schnittstellen wie auf dem Imote 2 nutzen kann. Somit ist sie ohne große Anpassungen sowohl im Simulator als auch auf echter Hardware lauffähig.

Abb. 5.17 zeigt die Architektur der Implementierung als UML-Klassendiagramm[13]. Die Anwendung besteht hier aus der Klasse *NodeApplication*, welche von der *Promid*-Klasse (Produktions-Middleware) erbt. Diese bietet der Anwendung Zugriff auf die Funktionalität der Middleware. Sie erbt selbst von den drei Klassen der BiPS-ns-3-Integration *BipsPtpMux*, *BipsTimer* und *BipsEvent*. Die Klasse *BipsPtpMux* erbt schließlich von der ns-3-Klasse *Application*. Somit handelt es sich bei der Anwendungsklasse *NodeApplication* ebenfalls um eine ns-3-Applikation. Allerdings nutzt sie die ns-3-Funktionen, beispielsweise zum Senden und Empfangen von Rahmen, nicht direkt, sondern greift ausschließlich auf die Promid-Middleware zu, welche wiederum auf die BiPS-ns-3-Integration zugreift. Diese nutzt schließlich die ns-3-Funktionalität. Auf dem Imote 2 würde anstatt der BiPS-ns-3-Integration das BiPS-Framework genutzt, welches auf die Hardware des Imote 2 zugreift.

[13] Das Diagramm zeigt nur eine Auswahl der vorhandenen privaten Methoden.

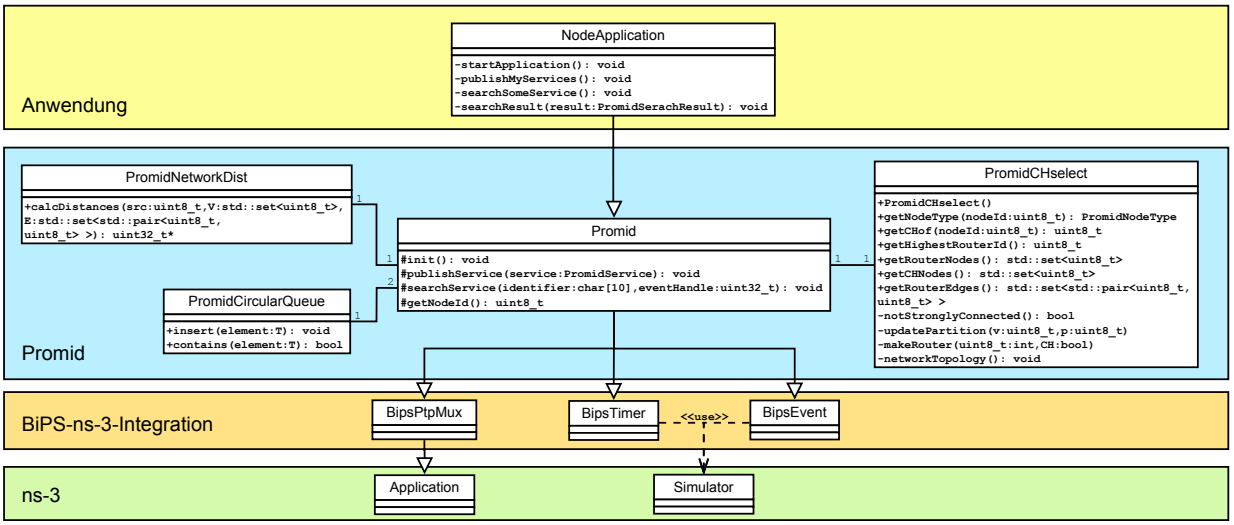


Abbildung 5.17.: UML-Klassendiagramm der Implementierung

Im ns-3-Simulator wird ein *Peer-to-Peer-Medium* und kein echtes Funkmedium simuliert. Dies bedeutet, dass die Topologie des Netzwerks statisch konfiguriert wird und sich die Topologie nicht durch die Simulation ergibt. Dadurch, dass die zu simulierende Topologie vorgegeben werden kann, können die resultierenden Ergebnisse besser mit den erwarteten Ergebnissen verglichen werden, was das Testen vereinfacht.

Auf jedem Knoten wird ein *NodeApplication*-Objekt erzeugt. Alle Knoten haben die selbe Architektur, das Verhalten der Middleware unterscheidet sich allerdings abhängig davon, ob der Knoten vom Clustering-Algorithmus zum CH, Gateway oder Follower gemacht wird. Die Methode *startApplication()* der Anwendungsklasse wird durch den ns-3-Simulator beim Starten eines Knotens aufgerufen. Sie initialisiert die Middleware, indem sie die *init()*-Methode der *Promid*-Klasse aufruft. Außerdem setzt sie zwei Timer. Der erste Timer ruft die *publishMyServices()*-Methode auf und publiziert damit die Dienste des Testknotens bei der Registry über die Promid-Methode *publishService()*. Diese Methode implementiert die in Kapitel 5.2.3 beschriebene Publikation eines Dienstes. Der zweite Timer ruft die *searchSomeService()*-Methode auf, welche die Promid-Methode *searchService()* nutzt, um nach einem Dienst zu suchen (s. Kapitel 5.2.4). Aktuell unterstützt die Middleware ausschließlich die Suche eines Dienstes anhand der Dienstkennung (*identifier*). Das Interface zur Suche ist asynchron, d.h. nachdem eine Suche gestartet wurde, läuft die Applikation weiter und wird informiert, sobald das Suchergebnis feststeht. Dabei triggert die Middleware die Ausführung des BiPS-Events mit dem Handle *eventHandle*, welches in der Beispielanwendung die *searchResult()*-Methode aufruft und ihr das Suchergebnis *result* übergibt. Weiterhin kann die Anwendung über die Promid-Methode *getNodeId()* die eigene Knoten-ID abfragen, beispielsweise um sie als ID des Service Providers für selbst bereitgestellte Dienste einzutragen.

Die Middleware besteht neben der *Promid*-Klasse aktuell aus drei weiteren Klassen. Die *PromidCHselect*-Klasse enthält den in Kapitel 5.2.2 vorgestellten Clustering-Algorithmus. Neben den Hilfsmethoden *notStronglyConnected()* (s. Listing 5.7), *makeRouter()* (s. Listing 5.8) und *updatePartition()* (s. Listing 5.9) enthält diese Klasse eine Methode *networkTopology()*, welche die Netzwerk-Topologie und die Konfiguration (V_{opt}, V_{obl} und V_{excl}) einliest. In der Simulation wird die in ns-3 konfigurierte Topologie angefragt. Auf dem Imote 2 würde hier das Ergebnis der automatischen Topologie-Bestimmung abgefragt. Der eigentliche Clustering-Algorithmus wird

durch den Konstruktor *PromidCHselect()* ausgeführt. Die Ergebnisse des Algorithmus können anschließend über die folgenden sechs Methoden abgerufen werden:

- *getNodeType(nodeId)* liefert zu einer gegebenen Knoten-ID den Knoten-Typen zurück (CH, Gateway oder Follower). Mit Hilfe dieser Methode bestimmt die Promid-Klasse u.a. den eigenen Knoten-Typ und verhält sich dementsprechend unterschiedlich.
- *getCHof(nodeId)* liefert die Knoten-ID des CH-Knotens, zu dem der Knoten mit der ID *nodeId* gehört. Mit dieser Methode fragt die Promid-Klasse u.a. ab, welcher Knoten der eigene CH ist, zu dem sie z.B. Publish- und Search-Nachrichten sendet.
- *getHighestRouterId()* liefert die höchste Router-ID. Diesen Wert nutzt Promid bei der Allokation von Speicher zur Bemessung von Arrays.
- *getRouterNodes()* gibt die gewählten Router-Knoten zurück (V_{router}).
- *getCHNodes()* gibt die gewählten CH-Knoten zurück (V_{CH}).
- *getRouterEdges()* gibt die Links des Router-Netzwerks zurück (E_{router}).

Die Klasse *PromidNetworkDist* dient dazu, die kürzesten Distanzen von einem Knoten *src* zu allen anderen Knoten in der übergebenen Topologie zu bestimmen. Da bisher noch kein Network Layer vorhanden ist, der diese Funktionalität bietet, wurde für die Middleware hier eine Implementierung des Dijkstra-Algorithmus integriert. Die Promid-Klasse nutzt diese Information bei der Verbreitung von Search-Nachrichten, um die Distanz zum ursprünglichen CH zu berechnen (s. Kapitel 5.2.4). Ohne Network Layer steht außerdem kein Routing zur Verfügung, um SearchResult-Nachrichten wie in Kapitel 5.2.4 beschrieben per Unicast zum ursprünglich suchenden Knoten zu übertragen. Daher erfolgt die Übertragung per Broadcast über das Router-Netzwerk. Dabei wird durch *PromidNetworkDist* die kürzeste Entfernung auf dem Router-Netzwerk zum ursprünglich suchenden Knoten bestimmt und der Broadcast auf diesen Radius beschränkt.

Die Klasse *PromidCircularQueue* implementiert einen Ringpuffer. Die Promid-Klasse nutzt diesen, um die Identifier der bereits weitergeleiteten Nachrichten sowie der bereits beantworteten Suchen zu speichern.

Die Klassen der BiPS-ns-3-Integration stellen der Middleware Funktionalität zum Senden und Empfangen von Rahmen sowie zum Nutzen von Timern und Events bereit und verwenden dabei in der Simulation u.a. die ns-3-Klassen *Application* und *Simulator*.

6. KAPITEL

Zusammenfassung und Ausblick

In der vorliegenden Masterarbeit wurden zunächst die Anforderungen an drahtlose Kommunikationssysteme im Produktionsbereich zusammengefasst und anhand eines Anwendungsszenarios konkretisiert. Dabei hat sich gezeigt, dass die Anforderungen in diesem Bereich sich von anderen Bereichen, in denen drahtlose Kommunikationssysteme eingesetzt werden, wie beispielsweise der Unterhaltungselektronik, deutlich unterscheiden: Hier stehen nicht in erster Linie Performanz-Aspekte wie hohe Übertragungsraten im Vordergrund, sondern Aspekte wie Zuverlässigkeit, Sicherheit, Energieeffizienz und die garantierte Einhaltung von Dienstgüte.

Daher ist es nicht verwunderlich, dass sich bei der Analyse von existierenden Standards gezeigt hat, dass aus anderen Bereichen stammende Standards nicht oder nur unzureichend den gesteckten Anforderungen gerecht werden können. Die näher betrachteten Standards WirelessHART und ISA 100.11a, welche speziell für den hier betrachteten Einsatzbereich entwickelt wurden, eignen sich dagegen erwartungsgemäß besser. Der direkte Vergleich dieser beiden Standards hat neben vielen grundlegenden Gemeinsamkeiten vor allem Unterschiede im Detail aufgezeigt. Insgesamt zeigt sich ISA 100.11a flexibler, wohingegen WirelessHART strengere Vorgaben macht und damit den Einsatz und die Kompatibilität der Geräte vereinfacht.

Obwohl das anschließend entworfene drahtlose Kommunikationssystem ebenfalls Gemeinsamkeiten mit WirelessHART und ISA 100.11a teilt, sieht es in vielen Punkten Lösungen vor, welche den Anforderungen des Anwendungsfalls besser gerecht werden. Auf MAC-Ebene werden exklusive Reservierungen flexibler Länge unterstützt, was eine bessere Ausnutzung des Mediums zur Folge hat. Außerdem ist die Superslot-Konfiguration während des Betriebs dynamisch anpassbar. Dies ermöglicht höheren Ebenen, Dienste dynamisch während des Betriebs zu abonnieren und abzubestellen. Die höhere Flexibilität geht dabei nicht zu Lasten der Zuverlässigkeit: Mit exklusiven Reservierungen und QoS-Unterstützung des Routings und auf Middleware-Ebene können Dienstgüte-Garantien zuverlässig eingehalten werden. Dank automatischer Topologie-Bestimmung müssen vorhandene Kommunikationslinks nicht manuell konfiguriert werden und anhand der erkannten Interferenzlinks wird die effiziente Nutzung von SDMA vereinfacht. Die Unterscheidung zwischen mobilen und stationären Knoten erlaubt es beispielsweise, mobile Roboter im selben Kommunikationssystem anzubinden wie stationäre Knoten, ohne dass sich für die Kommunikation zwischen stationären Knoten Nachteile ergeben. Weiterhin wird zwischen batteriebetriebenen Knoten und solchen mit einer festen Energieversorgung unterschieden, sodass erstere bestmöglich entlastet und ihre Batterielebensdauer erhöht werden kann.

Nicht alle Aspekte dieses Konzeptes konnten im Rahmen dieser Arbeit im Detail ausgearbeitet werden. Insbesondere die Unterstützung dynamischer Reservierungen sowie mobiler Knoten

wurde nur skizziert und bietet interessante Fragestellungen für weitere Arbeiten. Ein geeignetes Routing-Verfahren, welches mobile und batteriebetriebene Knoten berücksichtigt und trotzdem QoS-Unterstützung bietet, stellt einen weiteren Themenkomplex für anschließende Forschung dar.

Der Middleware Layer des konzipierten Kommunikationssystems wurde im Rahmen dieser Arbeit nicht nur konzeptionell entworfen, sondern in Teilen auch implementiert und im Simulator getestet. Dabei wurde anders als bei WirelessHART oder ISA 100.11a ein dienstorientierter Ansatz verfolgt. Dadurch können Dienste bei einer Service Registry dynamisch registriert und aufgesucht werden, sodass Dienstnutzer und Dienstanbieter entkoppelt werden. Das Austauschen von Knoten oder Verschieben von Diensten auf andere Knoten wird damit erleichtert und so die Flexibilität des Systems verbessert. Mit der partiell verteilten Service Registry wurde außerdem eine skalierbare Architektur implementiert. Indem der entwickelte Algorithmus zur Auswahl der Service Registry-Knoten ein Dominating Set bildet, wird sichergestellt, dass jeder Knoten einen schnellen Zugriff auf die Service Registry hat. Knoten ohne feste Stromversorgung können von der Wahl als Service Registry-Knoten ausgeschlossen werden, sodass sie durch den Nachrichtenaustausch, der zwischen Service Registry-Knoten stattfindet, nicht zusätzlich belastet werden. Durch die über einen Parameter konfigurierbare Replikation kann je nach Bedarf von einer vollständigen Replikation über eine lokal beschränkte Replikation bis hin zu keiner Replikation das System an das Nutzungsprofil der Anwendung angepasst werden.

Außer der Service Registry muss die Middleware auch Unterstützung für das Aufrufen, Abonnieren und Abbestellen von Diensten bieten. Diese Funktionalität, bei der insbesondere die QoS-Unterstützung wichtig ist, wurde nicht im Rahmen dieser Arbeit implementiert und wird im Anschluss daran fertiggestellt werden. Außerdem wurde im Rahmen dieser Arbeit die implementierte Service Registry ausschließlich im ns-3-Simulator getestet. Die Migration auf die Imote 2-Plattform steht noch aus. Sie sollte aber keinen großen Aufwand darstellen, da die Implementierung die gleichen Schnittstellen nutzt, die das BiPS-Framework auch auf dem Imote 2 zur Verfügung stellt.

A. ANHANG

Abschätzung der minimalen Datenrate

Im Folgenden soll die minimal benötigte Datenrate beispielhaft abgeschätzt werden, die ein drahtloses Kommunikationssystem bieten muss, um für den Einsatz zur Überwachung, Steuerung und Regelung von Produktionsanlagen geeignet zu sein. Dazu wird die minimale Datenrate berechnet, die benötigt wird, um alle Nachrichten aus dem Anwendungsszenario aus Kapitel 2 zu versenden und dabei die maximal erlaubte Verzögerung einzuhalten. Es wird für diese Abschätzung von einem TDMA-basierten Medienzugriff mit exklusiven Reservierungen ausgegangen. Für periodische Nachrichten ergibt sich das Intervall, in dem Slots reserviert werden, aus dem Minimum des Sendeintervalls und der maximalen Verzögerung. Bei ereignisbasierten Nachrichten ergibt sich das reservierte Intervall aus dem Minimum von maximaler Verzögerung und dem minimalen Ereignis-Eintritts-Intervall Int_{min}. Es wird bei der Berechnung der Netto-Datenrate vereinfachend davon ausgegangen, dass kein Overhead durch Synchronisation, Inter-Frame-Spaces oder Umschaltzeiten anfällt[14].

Die Topologie betreffend wird der ungünstigste Fall angenommen, bei dem jeder Knoten beim Senden alle anderen Knoten stört und daher keine gleichzeitigen Übertragungen stattfinden können. Multipliziert man für jeden Nachrichtentyp die Nachrichtengröße[15] mit der Anzahl einzuplanender Nachrichten je Sekunde, so erhält man in der Summe eine minimale Netto-Übertragungsrate von ca. 12,3 kbit/s (s. Tab. A.1) für einen Knoten. Geht man weiter davon aus, dass im Schnitt alle Nachrichten über 5 Hops weitergeleitet werden und daher 5 mal übertragen werden müssen, so ergibt sich eine minimale Netto-Datenrate von 61,5 kbit/s für das gesamte Netzwerk. Die benötigte Brutto-Datenrate unterscheidet sich z.B. auf Grund unterschiedlicher Header-Längen je nach Kommunikationssystem. Beim Einsatz von IEEE 802.15.4 ergibt sich beispielsweise auf dem Physical Layer ein Overhead von 6 Byte für Präambel, Start of frame Delimiter (SFD) und Längenfeld, dazu kommen mindestens 5 Byte durch Header des MAC Layers und weitere auf dem Network Layer. Da im Anwendungsszenario sehr kleine Nachrichten geschickt werden (max. 10 Byte), ergibt sich ein recht hoher Header-Anteil. Wenn man davon ausgeht, dass der Aufwand für Header und Netzwerk-Management zwei Drittel des gesamten Verkehrs ausmacht, so reicht eine Brutto-Datenrate von mindestens 185 kbit/s im Anwendungsszenario aus.

[14] Dieser Overhead wird bei der Abschätzung der Brutto-Datenrate berücksichtigt.
[15] Inkl. eindeutiger Kennung (16 Bit) sowie bei Sensorwerten Zeitstempel (32 Bit)

Tabelle A.1.: Abschätzung der Datenrate im betrachteten Anwendungsszenario

	Bit / Nachricht	Intervall	Anzahl	Reservierungen/s	Bit/s
Sensorwerte					
Temperatur	80	1,0	10	10,0	800,0
Druck	80	1,0	6	6,0	480,0
Gaserkennung	56	0,5	2	4,0	224,0
Vibration	56	1,0	10	10,0	560,0
Drehzahl	64	0,1	10	100,0	6.400,0
Lichtschranke	$49 \rightarrow 56$	1,0	10	10,0	560,0
Produktfertigstellung	56	5,0	1	0,2	11,2
Maschinenzustand	52	1,0	20	20,0	1040,0
Roboterposition	80	1,0	1	1,0	80,0
Steuerungsbefehle					
Ventilöffnung / Schließung	24	1,0	4	4,0	96,0
Maschinenabschaltung	$17 \rightarrow 24$	1,0	10	10,0	240,0
Zielposition Roboter	48	1,0	1	1,0	48,0
Heizungs-Solltemperatur	32	1,0	5	5,0	160,0
Regelungsbefehle					
Motorgeschwindigkeit	32	0,5	5	10,0	320,0
Ventilöffnung / Schließung	24	0,5	4	8,0	192,0
Maschinenabschaltung	$17 \rightarrow 24$	0,1	4	40,0	960,0
Heizungs-Solltemperatur	32	1,0	2	2,0	64,0
Gesamt					12.235,2

Literaturverzeichnis

[ABB] ABB: *ABB Group - Automation and Power Technologies*. `http://www.abb.com`.

[AGB11] AKERBERG, JOHAN, MIKAEL GIDLUND und MATS BJORKMAN: *Future research challenges in wireless sensor and actuator networks targeting industrial automation*. In: *9th IEEE International Conference on Industrial Informatics (INDIN)*, Seiten 410–415. IEEE, 2011.

[APVH00] AMIS, A.D., R. PRAKASH, T.H.P. VUONG und D.T. HUYNH: *Max-min d-cluster formation in wireless ad hoc networks*. In: *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, Band 1, Seiten 32–41 vol.1, 2000.

[AY07] ABBASI, AMEER AHMED und MOHAMED YOUNIS: *A survey on clustering algorithms for wireless sensor networks*. Computer communications, 30(14):2826–2841, 2007.

[Bak05] BAKER, NICK: *ZigBee and Bluetooth strengths and weaknesses for industrial applications*. Computing & Control Engineering Journal, 16(2):20–25, 2005.

[BBH13] BENTALEB, ABDELHAK, ABDELHAK BOUBETRA und SAAD HAROUS: *Survey of Clustering Schemes in Mobile Ad hoc Networks*. Communications and Network, 5:8, 2013.

[BGK13] BRAUN, TOBIAS, REINHARD GOTZHEIN und THOMAS KUHN: *Mode-Based Scheduling with Fast Mode-Signaling - A Method for Efficient Usage of Network Time Slots*. In: *6th International Conference on Computer Science and Information Technology (ICCSIT)*, 2013.

[Blu04] BLUETOOTH SIG: *Bluetooth Specification 2.0 +EDR*, November 2004.

[Blu09] BLUETOOTH SIG: *Bluetooth Specification 3.0 +HS*, April 2009.

[Blu10] BLUETOOTH SIG: *Bluetooth Specification 4.0*, Juni 2010.

[Blu13a] BLUETOOTH SIG: *Bluetooth Basics*. `http://www.bluetooth.com/Pages/Basics.aspx`, Dezember 2013.

[Blu13b] BLUETOOTH SIG: *Bluetooth Specification 4.1*, Dezember 2013.

[Blu13c] BLUETOOTH SIG: *Low Energy*. `http://www.bluetooth.com/Pages/Low-Energy.aspx`, Dezember 2013.

[Blu14] BLUETOOTH SIG: *About the Bluetooth SIG*. `https://www.bluetooth.org/en-us/members/about-sig`, Juni 2014.

[CAH96] CAMPBELL, ANDREW, CRISTINA AURRECOECHEA und LINDA HAUW: *A review of QoS architectures*. In: *Proc. of 4th IFIP International Workshop on Quality of Service (IWQS'96), Paris, France*, 1996.

[CGR12] CHRISTMANN, D., R. GOTZHEIN und S. ROHR: *The Arbitrating Value Transfer Protocol (AVTP) - Deterministic Binary Countdown in Wireless Multi-Hop Networks*. In: *Computer Communications and Networks (ICCCN), 2012 21st International Conference on*, Seiten 1 –9, August 2012.

[Chi07] CHIPCON / TEXAS INSTRUMENTS: *CC2420 datasheet*. `http://focus.ti.com/lit/ds/symlink/cc2420.pdf`, 2007. Revision SWRS041b.

[CNM10] CHEN, DEJI, MARK NIXON und ALOYSIUS MOK: *Application Layer*. In: *WirelessHART™*, Seiten 39–44. Springer, 2010.

[Cro09] CROSSBOW: *Imote 2 Datasheet*. `http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/Imote2_Datasheet.pdf`, 2009.

[DB97] DAS, B. und V. BHARGHAVAN: *Routing in ad-hoc networks using minimum connected dominating sets*. In: *IEEE International Conference on Communications, 1997. ICC '97 Montreal, Towards the Knowledge Millennium*, Band 1, Seiten 376–380 vol.1, Juni 1997.

[DH98] DEERING, STEPHEN und R. HINDEN: *RFC 2460 - Internet protocol, version 6 (IPv6) specification*. `http://tools.ietf.org/html/rfc2460`, Dezember 1998.

[DW05] DAI, FEI und JIE WU: *On Constructing k-Connected k-Dominating Set in Wireless Networks*. In: *Parallel and Distributed Processing Symposium, 2005. Proceedings. 19th IEEE International*, Seiten 81a–81a, April 2005.

[ECG14] ENGEL, MARKUS, DENNIS CHRISTMANN und REINHARD GOTZHEIN: *Implementation and Experimental Validation of Timing Constraints of BBS*. In: *Wireless Sensor Networks*, Seiten 84–99. Springer, 2014.

[Eis06] EISENACHER, MICHAEL: *Optimierung von Ultra-Wideband-Signalen (UWB)*. Doktorarbeit, Universität Karlsruhe, 2006.

[Ene02] ENERGETICS INC. (IN COLLABORATION WITH US DEPT. OF ENERGY, OFFICE OF ENERGY EFFICIENCY AND RENEWABLE ENERGY): *Industrial wireless technology for the 21st century*. Proc. Industrial Wireless Workshop, Dezember 2002.

[Ene11] ENERGY HARVESTING JOURNAL: *Devawave tapes out scensor DW1000 chip*. `http://www.decawave.com/assets/files/press/Energy%20Harvesting%20Journal%20July%2028%202011.pdf`, Juli 2011.

[Eng13] ENGEL, MARKUS: *Optimierung und Evaluation Black Burst-basierter Protokolle unter Verwendung der Imote 2-Plattform*. Masterarbeit, TU Kaiserslautern, 2013.

[Erl05] ERL, THOMAS: *Service-Oriented Architecture: Concepts, Technology, and Design*. Prentice Hall, 2005.

[Eur10] EUROPEAN COMMITTEE FOR ELECTROTECHNICAL STANDARDIZATION (CENELEC): *Industrial communication networks - Wireless communication network and communication profiles - WirelessHART™ (IEC 62591:2010)*, Juni 2010.

[GH09] GUNGOR, VEHBI C und GERHARD P HANCKE: *Industrial wireless sensor networks: Challenges, design principles, and technical approaches*. Industrial Electronics, IEEE Transactions on, 56(10):4258–4265, 2009.

[GK08] GOTZHEIN, REINHARD und THOMAS KUHN: *Decentralized Tick Synchronization for Multi-Hop Medium Slotting in Wireless Ad Hoc Networks Using Black Bursts*. In: *Proceedings of the Fifth Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks, SECON 2008, June 16-20, 2008, Crowne Plaza, San Francisco International Airport, California, USA*, Seiten 422–431. IEEE, 2008.

[GOP12] GOMEZ, CARLES, JOAQUIM OLLER und JOSEP PARADELLS: *Overview and Evaluation of Bluetooth Low Energy: An Emerging Low-Power Wireless Technology*. Sensors, 12(9):11734–11753, 2012.

[HA06] HANCKE, GERHARD P und BEN ALLEN: *Ultrawideband as an industrial wireless solution*. Pervasive Computing, IEEE, 5(4):78–85, 2006.

[Haa98] HAARTSEN, JAAP: *Bluetooth-The universal radio interface for ad hoc, wireless connectivity*. Ericsson review, 3(1):110–117, 1998.

[HAR10] HART COMMUNICATION FOUNDATION: *WirelessHART Approved by IEC as First International Standard for Wireless Communication in Process Automation*. http://hartcomm.org/hcf/news/pr2010/WirelessHART_approved_by_IEC.html, April 2010.

[HAR14a] HART COMMUNICATION FOUNDATION: *About the HART Protocol*. http://www.hartcomm.org/protocol/about/aboutprotocol.html, Januar 2014.

[HAR14b] HART COMMUNICATION FOUNDATION: *HART Communication Foundation Product Catalog*. http://www.hartcommproduct.com/inventory2/index.php?action=listcat, Mai 2014.

[Hei09] HEISE NETZE: *Bluetooth SIG wendet sich von UWB ab*. http://www.heise.de/netze/meldung/Bluetooth-SIG-wendet-sich-von-UWB-ab-847114.html, Oktober 2009.

[HT11] HUI, JONATHAN und PASCAL THUBERT: *RFC 6282 - Compression format for IPv6 datagrams over IEEE 802.15. 4-based networks*. http://tools.ietf.org/html/rfc6282, September 2011.

[IEC10] IEC: *IEC 62591 ed1.0*, April 2010.

[IEC11] IEC: *IEC 62601 ed1.0*, November 2011.

[IEC12] IEC: *IEC/PAS 62734 ed1.0*, März 2012.

[IEC14] IEC: *IEC 61158-1 ed1.0*, Mai 2014.

[IEE99] IEEE: *IEEE 802.11b™-1999*, September 1999.

[IEE03a] IEEE: *IEEE 802.11a™-1999*, Juni 2003.

[IEE03b] IEEE: *IEEE 802.11g™-2003*, Juni 2003.

[IEE03c] IEEE: *IEEE 802.15.4™-2003*, Oktober 2003.

[IEE04a] IEEE: *IEEE 802.11i™-2004*, Juli 2004.

[IEE04b] IEEE: *IEEE 802.15 WPAN High Rate Alternative PHY Task Group 3a (TG3a)*. http://www.ieee802.org/15/pub/TG3a.html, August 2004.

[IEE05a] IEEE: *IEEE 802.11e™-2005*, November 2005.

[IEE05b] IEEE: *IEEE 802.15.1™-2005*, Juni 2005.

[IEE06] IEEE: *IEEE 802.15.4™-2006*, Septeber 2006.

[IEE07] IEEE: *IEEE 802.15.4a™-2007*, August 2007.

[IEE09a] IEEE: *IEEE 802.11n™-2009*, Oktober 2009.

[IEE09b] IEEE: *IEEE 802.15.4c™-2009*, April 2009.

[IEE09c] IEEE: *IEEE 802.15.4d™-2009*, April 2009.

[IEE11a] IEEE: *IEEE 802.15.4™-2011*, September 2011.

[IEE11b] IEEE: *IEEE P802.11 TGs*. http://grouper.ieee.org/groups/802/11/Reports/tgs_update.htm, Juli 2011.

[IEE12a] IEEE: *IEEE 802.11™-2012*, März 2012.

[IEE12b] IEEE: *IEEE 802.11ad™-2012*, Oktober 2012.

[IEE12c] IEEE: *IEEE 802.15.4e™-2012*, April 2012.

[IEE12d] IEEE: *IEEE 802.15.4f™-2012*, April 2012.

[IEE13a] IEE: *Wireless: Keine Einigung möglich*. IEE Elektrische Automatisierung + Antriebstechnik, 58:8, April 2013.

[IEE13b] IEEE: *IEEE P802.11 - TASK GROUP AC*. http://grouper.ieee.org/groups/802/11/Reports/tgac_update.htm, September 2013.

[igr] IGRAPH CORE TEAM: *igraph R package*. http://igraph.org/r/.

[IME07] IMEC: *Ultra-low-power radio*. http://www.imec.be/ScientificReport/SR2007/html/1384152.html, 2007.

[Int94] INTERNATIONAL TELECOMMUNICATION UNION (ITU): *Information technology – Open Systems Interconnection – Basic Reference Model: The basic model*, Juli 1994.

[Int07a] INTERNATIONAL ORGANIZATION FOR STANDARDIZATION: *ISO/IEC 26907:2007, Information technology – Telecommunications and information exchange between systems – High Rate Ultra Wideband PHY and MAC Standard*, 2007.

[Int07b] INTERNATIONAL ORGANIZATION FOR STANDARDIZATION: *ISO/IEC 26908:2007, Information technology – Telecommunications and information exchange between systems – MAC-PHY Interface for ISO/IEC 26907*, 2007.

[ISA] ISA: *ISA | The International Society of Automation*. http://www.isa.org/.

[ISA08] ISA: *The ISA100 Standards - Overview and Status*. http://www.isa.org/Content/Microsites1134/SP100,_Wireless_Systems_for_Automation/Home1034/ISA100_Overview_Oct_2008.pdf, Oktober 2008.

[ISA11] ISA: *ANSI/ISA-100.11a-2011 Wireless systems for industrial automation: Process control and related applications*, Mai 2011.

[ISA14a] ISA: *About ISA*. http://www.isa.org/Content/NavigationMenu/General_Information/About_ISA1/About_ISA.htm, Januar 2014.

[ISA14b] ISA: *Automation Standards Compliance Institute (ASCI)*. https://www.isa.org/standards-and-publications/isa-standards/automation-standards-compliance-institute/, Mai 2014.

[ISA14c] ISA-100 WIRELESS COMPLIANCE INSTITUTE: *ISA-100 Wireless Compliance Institute*. http://www.isa100wci.org/, Mai 2014.

[ISA14d] ISA-100 WIRELESS COMPLIANCE INSTITUTE: *ISA100 Wireless Product Listing*, Mai 2014.

[Ive07] IVERSEN, WES: *WirelessHart Ready for Prime Time*. http://www.automationworld.com/information-management/wirelesshart-ready-prime-time, Oktober 2007.

[Kow02] KOWALK, W.: *Rechnernetze - Dezentrale Zuteilungsprotokolle*. http://einstein.informatik.uni-oldenburg.de/rechnernetze/seite211.htm, März 2002.

[Kra13] KRAMER, CHRISTOPHER: *Ermittlung des Netzzustands von Funk-Netzwerken*. http://vs.informatik.uni-kl.de/publications/2013/Kr13/Kr13.pdf, 2013.

[LLYL13] LIANG, WEI, SHUAI LIU, YUTUO YANG und SHIMING LI: *Research of Adaptive Frequency Hopping Technology in WIA-PA Industrial Wireless Network*. In: *Advances in Wireless Sensor Networks*, Seiten 248–262. Springer, 2013.

[LSH08] LENNVALL, TOMAS, STEFAN SVENSSON und FREDRIK HEKLAND: *A comparison of WirelessHART and ZigBee for industrial applications*. In: *IEEE International Workshop on Factory Communication Systems, 2008. WFCS 2008.*, Seiten 85–88. IEEE, 2008.

[LWE05] LOW, KAY-SOON, W.N.N. WIN und MENG-JOO ER: *Wireless Sensor Networks for Industrial Environments*. In: *International Conference on Computational Intelligence for Modelling, Control and Automation, 2005 and International Conference on Intelligent Agents, Web Technologies and Internet Commerce*, Band 2, Seiten 271–276, Nov 2005.

[MTA05] MATHIESEN, MOGENS, GILLES THONET und NIELS AAKWAAG: *Wireless ad-hoc networks for industrial automation: current trends and future prospects*. In: *Proceedings of the IFAC World Congress, Prague, Czech Republic*, Seiten 89–100, 2005.

[ns-] NS-3 PROJEKT: *ns-3*. http://www.nsnam.org/.

[Pos80] POSTEL, JON: *RFC 768 - User datagram protocol*. http://tools.ietf.org/html/rfc768, August 1980.

[PRML06] PETROVA, M., J. RIIHIJARVI, P. MAHONEN und S. LABELLA: *Performance study of IEEE 802.15.4 using measurements and simulations*. In: *Wireless Communications and Networking Conference, 2006. WCNC 2006. IEEE*, Band 1, Seiten 487–492, April 2006.

[PW10] PATEL, MAULIN und JIANFENG WANG: *Applications, challenges, and prospective in emerging body area networking technologies*. Wireless Communications, IEEE, 17(1):80–88, 2010.

[SS12] SCHILL, ALEXANDER und THOMAS SPRINGER: *Verteilte Systeme*. Springer-Verlag Berlin Heidelberg, 2012.

[Staa] STATISTA / IDC: *Absatz von Smartphones weltweit vom 1. Quartal 2009 bis zum 1. Quartal 2014*. http://de.statista.com/statistik/daten/studie/246300/umfrage/weltweiter-absatz-von-smartphone-nach-quartalen/.

[Stab] STATISTA / IDC: *Absatz von Tablets weltweit in den Jahren 2010 bis 2013*. http://de.statista.com/statistik/daten/studie/253303/umfrage/weltweiter-absatz-von-media-tablets/.

[Stac] STATISTA / JIWIRE: *Anzahl der öffentlichen Wi-Fi Locations und Hot Spots weltweit vom 2. Quartal 2009 bis zum 2. Quartal 2013*. http://de.statista.com/statistik/daten/studie/158346/umfrage/anzahl-der-wi-fi-locations-weltweit-seit-dem-2-quartal-2009/.

[US 06] US DEPT. OF ENERGY, OFFICE OF ENERGY EFFICIENCY AND RENEWABLE ENERGY: *Wireless Sensor Network - Advanced Energy Management Solution for Industrial Motors*. August 2006.

[UWB06] UWB FORUM: *UWB Forum and WiMedia Alliance Committed to Commercializing UWB*. http://web.archive.org/web/20061006143558/http://www.uwbforum.org/index.php?option=com_content&task=view&id=121&Itemid=2, Januar 2006.

[Vol96] VOLKMANN, LUTZ: *Fundamente der Graphentheorie*. Springer Verlag Wien, 1996.

[WG07] WEBEL, CHRISTIAN und REINHARD GOTZHEIN: *Formalization of Network Quality-of-Service Requirements*. In: DERRICK, JOHN und JÜRI VAIN (Herausgeber): *FORTE*, Band 4574 der Reihe *Lecture Notes in Computer Science*, Seiten 309–324. Springer, 2007.

[WL99] WU, JIE und HAILAN LI: *On Calculating Connected Dominating Set for Efficient Routing in Ad Hoc Wireless Networks*. In: *Proceedings of the 3rd International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, DIALM '99, Seiten 7–14, New York, NY, USA, 1999. ACM.

[YHE02] YE, WEI, JOHN HEIDEMANN und DEBORAH ESTRIN: *An Energy-Efficient MAC Protocol for Wireless Sensor Networks*. In: *INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, Band 3, Seiten 1567–1576. IEEE, 2002.

[Zig08] ZIGBEE STANDARDS ORGANIZATION: *ZigBee Specification*. http://zigbee.org/Specifications/ZigBee/download.aspx, Januar 2008.

[ZLY09] ZHANG, XIAOLING, WEI LIANG und HAIBIN YU: *Adaptive Timeslot Scheduling of Long Cycle Data in WIA-PA network*. In: *Asia-Pacific Conference on Computational Intelligence and Industrial Applications, 2009. PACIIA 2009*, Band 1, Seiten 267–271, Nov 2009.