Universiteit Gent
Faculteit Ingenieurswetenschappen en Architectuur
Vakgroep Informatietechnologie

Promotoren: prof. dr. ir. Ingrid Moerman
dr. ir. Jeroen Hoebeke

Universiteit Gent
Faculteit Ingenieurswetenschappen en Architectuur

Vakgroep Informatietechnologie
Gaston Crommenlaan 8 bus 201, B-9050 Gent, België

Tel.: +32-9-331.49.00
Fax.: +32-9-331.48.99

Dit werk kwam tot stand in het kader van een specialisatiebeurs van het IWT-Vlaanderen (Instituut voor de aanmoediging van Innovatie door Wetenschap en Technologie in Vlaanderen).

Proefschrift tot het behalen van de graad van Doctor in de Ingenieurswetenschappen: Elektrotechniek
Academiejaar 2013-2014

Dankwoord

Eigenlijk is het al veel te laat op de avond om dit dankwoord nog te schrijven, maar het is zoals ze wel eens zeggen, de laatste loodjes wegen het zwaarst. Met het schrijven van dit dankwoord, sluit ik een periode van bijna 7 jaar boeiend onderzoek af. Het is het einde van een tijdperk dat soms meer weg had van een rollercoaster dan van een doctoraat. Daarom wil ik iedereen bedanken, die mij van ver of dichtbij gedurende al die jaren gesteund en gemotiveerd heeft.

In de eerste plaats wil ik mijn promotoren, prof. Ingrid Moerman en Jeroen Hoebeke, bedanken voor de gekregen kansen en voor het vertrouwen doorheen dit doctoraat. Het is een grote vakgroep, maar toch wisten ze er steeds opnieuw voor te zorgen dat ik er na de juiste feedback en waardevol advies weer met volle moed tegenaan kon gaan. Verder wil ik ook prof. Piet Demeester, voorzitter van de onderzoeksgroep IBCN, bedanken dat ik deel mocht uitmaken van 'zijn' vakgroep. Ik kijk terug op een leuke periode, waar naast werk, ook plezier kon gemaakt worden. En gelukkig zaten de proffen in onze vakgroep niet in hun ivoren toren, maar kon je in de wandelgangen ook gewoon een praatje met hen maken: bedankt Tom, Bart, Eric en alle anderen.

Graag wil ik ook de INTEC vakgroepvoorzitter, prof. Daniël De Zutter, zijn voorganger prof. Paul Lagasse en de huidig decaan prof. Rik Van de Walle bedanken om alles op de goede rails te houden achter de schermen. Daarnaast wil ik ook het agentschap voor Innovatie door Wetenschap en Technologie (IWT) bedanken dat gedurende 4 jaar gezorgd heeft voor de nodige financiering van mijn onderzoek.

Een doctoraat maak je niet alleen. Ik zou dan ook in de eerste plaats alle collega's van de 'mobile' groep willen bedanken voor de vele interessante discussies en hun soms verhelderende kijk op de zaak. Ook een bedankje aan de vele mensen van de Universiteit Antwerpen, de Vrije Universiteit Brussel en de Katholieke Universiteit Leuven waarmee ik gedurende de voorbije jaren mocht samenwerken.

Onkostennota's, reisaanvragen en beursaanvragen: bedankt Martine, Davinia, Karien, Ilse, Bernadette, Dalila, Joke en alle anderen om het leven administratief heel wat te vereenvoudigen. Ook het it-admin team mag niet ontbreken in dit dankwoord: bedankt Jonathan, Joeri, Bert, Brecht en al jullie voorgangers om alles technisch in goede banen te leiden. Bedankt ook Sandra en Sabrina, voor de wilde verhalen en het proper houden van ons gebouw.

Graag zou ik ook de mensen willen bedanken waar ik toch een groot deel van mijn doctoraat heb bij vertoefd, mijn bureaugenoten. In 2007 kwam ik tijdelijk in bureau 2.20 terecht, in afwachting van mijn definitieve plaats. Femke O.,

Bert, Jeroen en Samuel, bedankt voor de korte maar leuke periode. Daarna ben ik neergestreken in bureau 3.13 waar ik een hele leuk tijd heb gekend dankzij Lieven, Lien, Bart, Koen, Peter, Dimitri, Abram, Ruth, Mathieu, Ward, Tim, Stijn en Farhan. De 'grote bureauverhuis' heeft mij vervolgens naar bureau 3.15 gebracht. Samen met bureaugenoten Lieven en Bart kwam ik in een bureau terecht waar ik nog vaak zal aan terugdenken. Bedankt Pieter B, Pieter DM, Peter R, Stefan, Peter DV, Vincent, Jono, Frank, Eli en George.

Een aantal mensen wil ik graag in het bijzonder vermelden. In de eerste plaats is dit Lieven, die gelijktijdig met mij aan een doctoraat is begonnen en dit ook ongeveer gelijktijdig met mij heeft afgerond. Bedankt voor de vele gesprekken en het delen van lief en leed. Daarnaast mag ook Lien zeker niet vergeten worden. In deze mannenwereld voelden we ons soms wel eens de vreemde eend in de bijt, het deed dan ook deugd om tijdens de lunch eens over iets anders dan computers en allerhande technische zaken te kunnen praten. Je leerde me dat waar een wil is ook een weg is. Bedankt voor de steun in de moeilijkere periodes en de vele gesprekken over onze dochtertjes. Ook Koen mag hier niet ontbreken: bedankt voor de gesprekken bij de koffie en om mij met raad en daad bij te staan tijdens de laatste fase van mijn doctoraat. Tot slot ook nog een speciaal bedankje aan Bart om mij wegwijs te maken in de wondere wereld van het testbed en voor de lekkere chocomoussekes, en aan Pieter B en Pieter DM om te zorgen voor de vrolijke noot in onze bureau.

De boog kan natuurlijk niet altijd gespannen staan. Daar waren we in bureau 3.15 (en ook wel 3.13) goed in. Van verlovingsdrinks over huwelijksfeesten tot babyborrels. De vele feestjes zorgden voor de nodige ontspanning en de nieuwe generatie staat al klaar: Jasper, Lukas, Lisa, Finneke, Louis, Mona, Lara, Lotte, Mathis, Hanne, Lars, Arwen, Lise en Nathan.

Verder wil ik nog alle collega's bedanken die ik tegenkwam tijdens de lunch, bij een koffie, of gewoon in de wandelgangen en die mijn verblijf bij IBCN aangenamer maakten. Bedankt Martine voor de vele leuke babbels bij onze boterhammetjes, en ook bedankt aan de andere lunch-collega's die hun best deden om het niet-technisch te houden: Jonathan, Joeri, Koen, Bart, Pieter, Stefan en Eric. Bedankt ook aan Nicolas, mede op zijn aanraden ben ik aan een doctoraat begonnen, en gelukkig kwam ik hem regelmatig tegen in de wandelgangen om hierover mijn beklag te doen als het even wat minder ging ;-).

Ook buiten IBCN zijn er heel wat vrienden die mij positief geïnspireerd hebben en gezorgd hebben voor de nodige afleiding. Bedankt allemaal.

Rest mij nog de mensen te bedanken die het belangrijkste zijn in mijn leven, mijn familie. Bedankt ma en pa voor de gekregen kansen, zonder jullie had dit boek er niet geweest. Ook bedankt voor de vele etentjes en de helpende hand bij allerhande huishoudelijke taakjes en de bouw van ons droomhuis. Ook bedankt aan mijn broer Davy en schoonzus Sophie. We hebben de voorbije jaren heel wat leuke uitstapjes gemaakt. Ook mijn schoonouders verdienen een speciaal bedankje. Dankzij hen kon ik met een gerust gemoed aan mijn doctoraat werken omdat ik wist dat mijn twee lieve schatten, Lotte & Hanne, bij hen in goede handen waren. Verder nog bedankt aan mijn schoonbroer Karel, schoonzus Valérie en hun

dochtertje Louisa, ons weekendje Efteling was onvergetelijk en moeten we zeker nog eens overdoen met jullie zoontje - en tevens mijn metekindje - Robin erbij. Tot slot wil ik ook nog mijn grootmoeder Yvonne en mijn schoongrootouders Robert, Agnes en Marie-José bedanken voor hun interesse en de warme ontvangst bij elk bezoek.

Save the best for last:

Bedankt Maarten, mijn 'ventje', om inmiddels 10 jaar aan mijn zijde te staan. De plaats ontbreekt om je voor alles te bedanken wat je voor mij doet. Geen SQL, Matlab, PHP of debug-vraag was je tijdens dit doctoraat teveel, ook al had de klok al lang middernacht geslagen. Je brengt het beste in mij naar boven. Bedankt voor je hulp, liefde, steun, voor gewoonweg alles.

Bedankt Lotte & Hanne, mijn twee fantastische dochters van inmiddels 4 en bijna 2 jaar. Jullie hebben me leren relativeren en maken elke dag opnieuw de moeite waard. Jullie glimlach laat alle zorgen spontaan verdwijnen en maakt duidelijk wat echt belangrijk is.

Gent, juni 2014
Evy Troubleyn

Table of Contents

List of Figures

List of Tables

List of Acronyms

0-9

6LoWPAN	IPv6 over Low power Wireless Personal Area Networks

A

ACK	Acknowledgment
AODV	Ad hoc On-Demand Distance Vector

C

CCA	Clear Channel Assessment
CoAP	Constrained Application Protocol
CRC	Cyclic Redundancy Check
CSMA/CA	Carrier Sense Multiple Access/Collision Avoidance
CTP	Collection Tree Protocol
CTS	Clear To Send

D

DEUS	Deployment and Easy Use of wireless Services
DoA	Degree of Aggregation
DYMO	Dynamic MANET On-Demand

I

IDRA	Information Driven Architecture

IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IoT	Internet of Things
IP	Internet Protocol

K

kbps	kilobit per second

L

LPL	Low Power Listening

M

M2M	Multipoint-to-Multipoint
M2P	Multipoint-to-Point
MAC	Medium Access Control

O

OS	Operating System

P

P2M	Point-to-Multipoint
P2P	Point-to-Point

Q

QoS	Quality of Service

R

RERR	Route Error
RFID	Radio Frequency Identification
RREP	Route Reply
RREQ	Route Request
RSSI	Received Signal Strength Indicator
RTS	Request To Send

T

TDMA	Time Division Multiple Access
TI	Texas Instruments

U

UAV	Unmanned Aerial Vehicle
UDP	User Datagram Protocol
URI	Uniform Resource Identifier

W

WSN	Wireless Sensor Network
WiFi	Wireless Fidelity

Samenvatting
– Summary in Dutch –

Draadloze sensornetwerken zitten in de lift. Ze zijn opgebouwd uit vele goedkope en kleine autonome toestellen die draadloos met elkaar verbonden zijn. Traditioneel werden ze vooral gebruikt om omgevingen te monitoren, om processen op te volgen en te controleren, en voor industriële automatisering. Tegenwoordig maken ze ook deel uit van het Internet der Dingen. In dit Internet der Dingen worden alle toestellen en objecten uit het dagelijkse leven verbonden met het internet. We denken hierbij aan koelkasten, intelligente telefoons, auto's, maar ook aan gordijnen, kledij en zoveel meer. Cisco verwacht dat tegen eind 2020 meer dan 50 miljard toestellen en objecten met elkaar verbonden zullen zijn. Toepassingsdomeinen zijn de gezondheidszorg, het intelligente elektriciteitsnetwerk, het automatiseren van gebouwen, transport en logistiek en het intelligent monitoren van omgevingen, industrie en landbouw. De verbinding tussen de verschillende toestellen en objecten die deel uitmaken van het Internet der Dingen kan tot stand komen via WiFi, bluetooth, RFID, maar tegenwoordig gebeurt dit ook door het inschakelen van draadloze sensornetwerken. IP over sensornetwerken is intussen gestandaardiseerd, zodat draadloze sensornetwerken naadloos kunnen geïntegreerd worden met het Internet der Dingen.

Bij het uitbouwen van draadloze sensornetwerken is het zuinig omgaan met energie erg belangrijk. Draadloze sensoren hebben vaak een beperkte levensduur omdat ze gevoed worden door batterijen of zonnecellen, waardoor de energievoorraad steeds beperkt is. Het is dus van belang dat deze sensoren zo optimaal mogelijk functioneren om te kunnen genieten van een zo lang mogelijke levensduur. Het vervangen van batterijen is niet alleen zeer tijdrovend, maar in sommige omgevingen zelfs helemaal niet mogelijk. Het uitvallen van de batterij zorgt dan definitief voor het verdwijnen van het 'ding'.

Draadloze communicatie is van nature uit onbetrouwbaar. Wanneer draadloze sensornetwerken gekoppeld worden met het Internet is het dus belangrijk dat voldoende kwaliteit kan aangeboden worden. Ze worden immers gebruikt voor veel verschillende en vaak veeleisende toepassingen. Voor sommige toepassingen moeten de gegevens binnen een bepaalde tijd toekomen, terwijl voor andere toepassingen de gegevens foutloos en betrouwbaar moeten toekomen. Bovendien kunnen verschillende toepassingen terzelfdertijd over hetzelfde sensornetwerk uitgevoerd worden, elk met hun specifieke vereisten: een telefoongesprek dat slechts een beperkte vertraging en een minimum aan pakketverlies toelaat kan simul-

taan uitgevoerd worden met een belangrijke gegevensoverdracht die niet tijdsgelimiteerd is, maar wel een hoge betrouwbaarheid vereist.

Het uitgangspunt van dit proefschrift is om bij te dragen tot het succes van het Internet der Dingen door bewust om te gaan met energie en terzelfdertijd voldoende kwaliteit aan te bieden aan diverse toepassingen.

In de eerste fase van dit proefschrift gaan we dieper in op draadloze sensornetwerken en de onderzoeksuitdagingen die gekoppeld zijn aan het bieden van voldoende kwaliteitsgaranties. Na een inleidend hoofdstuk worden in Hoofdstuk 2 deze onderzoeksuitdagingen vertaald in ontwerpsvereisten en wordt een universeel raamwerk ontworpen dat aan de uiteenlopende vereisten van verschillende gelijktijdige applicaties tegemoet komt. We maken hierbij een onderscheid tussen kwaliteitsverlening op protocolniveau en op architectuurniveau. In Hoofdstuk 3 wordt dit raamwerk enerzijds geïmplementeerd en geëvalueerd in een bestaande netwerk simulator, en anderzijds geïntegreerd en gevalideerd in een bestaande sensornetwerk-architectuur op echte hardware in een testbed.

In de tweede fase van dit proefschrift wordt dieper ingegaan op het verbeteren van het energieverbruik in sensornetwerken die gekoppeld zijn met het Internet der Dingen. Een veelgebruikte techniek om energie te besparen in draadloze sensornetwerken is pakketaggregatie. Bij aggregatie worden vele verschillende kleine pakketjes samengevoegd tot één groot pakket. Het idee hierachter is dat hoe minder pakketten er verstuurd worden, hoe minder energie er verbruikt wordt. Dit komt doordat een sensornode voornamelijk voor de draadloze communicatie zeer veel energie verbruikt, nl. voor het verzenden en ontvangen van pakketten. Een sensornode die geen pakketten hoeft te verzenden of ontvangen kan zijn radio tijdelijk uitschakelen waardoor er nauwelijks nog energie wordt verbruikt. Helaas heeft aggregatie vaak een negatief effect op de aangeboden kwaliteit. Aggregeren betekent immers wachten op pakketten, en dus ook meer vertragingen. Daarom bestuderen we in Hoofdstuk 4 de impact van aggregatie op het energieverbruik en op verschillende kwaliteitsparameters zoals vertraging en betrouwbaarheid. Bovendien onderzoeken we hoe aggregatie geïmplementeerd kan worden binnen de IP standaard die gebruikt voor het Internet der Dingen en binnen het in Hoofdstuk 2 en 3 ontwikkelde dienstverleningsraamwerk.

Bij onze studie maken we een onderscheid tussen drie aggregatiescenario's: verkeer dat vanuit sensornetwerken via één of meerdere gateways naar het Internet gerouteerd wordt, verkeer tussen sensor nodes onderling, en verkeer dat van het Internet naar één of meerdere sensoren binnen een sensornetwerk gerouteerd wordt.

In Hoofdstuk 5 bekijken we het verkeer dat vanuit sensornetwerken naar het Internet gerouteerd wordt. Hierbij hebben we vooral te maken met de veel-naar-één relatie, waarbij het verkeer komende van verschillende sensornodes gerouteerd wordt naar een gateway. Hierbij zijn vaak veel aggregatiemogelijkheden aangezien vele knopen een gemeenschappelijk pad hebben naar de gateway, wat kan leiden tot een grote besparing op vlak van energie. Meer specifiek gaan we in dit hoofdstuk een mechanisme ontwikkelen dat in staat is om het niveau van aggregatie dynamisch aan te passen aan de actuele toepassing om steeds de meest optimale

kwaliteit te kunnen leveren. Er wordt hierbij steeds een afweging gemaakt tussen het verbruikte energieniveau en de graad van aangeboden dienstverlening.

In Hoofdstuk 6 analyseren we het verkeer dat tussen de sensor nodes onderling verloopt. Hierbij zijn er heel wat minder aggregatiemogelijkheden, wat nefast is voor het energieverbruik. Het verkeer verloopt vaak niet via één pad, maar er zijn verbindingen tussen veel verschillende sensor nodes. Daar waar in Hoofdstuk 5 de nadruk lag op het verbeteren van bestaande unicast aggregatietechnieken, waarbij enkel pakketten met dezelfde bestemming worden geaggregeerd, onderzoeken we in dit hoofdstuk het gebruik van broadcast aggregatie. Bij broadcast aggregatie wordt niet gewacht op pakketten met dezelfde bestemming vooraleer te aggregeren in een groter pakket, maar worden pakketten geaggregeerd ongeacht hun specifieke bestemming. Het resulterende geaggregeerde pakket wordt vervolgens als broadcast pakket verstuurd, waarna de ontvangende nodes inspecteren of dit geaggregeerde pakket een pakketje bevat dat voor hen bestemd was. Indien dit het geval is, dan wordt het desbetreffende pakket uit het aggregatiepakket gefilterd.

Tot slot bekijken we in Hoofdstuk 7 het scenario waarbij verkeer van het Internet naar één of meerdere sensoren binnen een sensornetwerk gerouteerd wordt. Ook hier zal standaard unicast aggregatie niet leiden tot een optimaal energieverbruik, omdat pakketten bijna altijd verschillende bestemmingen zullen hebben. In dit hoofdstuk zullen we onderzoeken of broadcast aggregatie tot energiewinst kan leiden.

In Hoofdstuk 8 worden de verworven inzichten van dit doctoraatsonderzoek samengevat en worden een aantal krijtlijnen voor toekomstig onderzoek uitgezet.

Summary

Wireless sensor networks are booming. They contain many small and cheap autonomous devices that are communicating through a wireless interface. Traditionally, they were mainly used for environmental monitoring, for process control and, and for industrial automation. Today they are also part of the Internet of Things (IoT). In this Internet of Things, all devices from everyday life are connected to the Internet. We can think of refrigerators, smartphones, cars, but even on curtains, clothes and many more. Cisco expects that by the end of 2020, more than 50 billion devices and objects will be interconnected with the Internet. Example applications are health care, smart grid and building automation, transport and logistics, and intelligent monitoring of environments, industry and agriculture. The connection between the objects and devices of the Internet of Things can be realized via WiFi, Bluetooth, RFID, but today, this is also done by using wireless sensor networks. IP over sensor networks became a standard, so it can be integrated seamlessly with the Internet of Things.

By deploying wireless sensor networks, economical use of energy consumption is very important. Wireless sensors often have a limited lifetime because they are powered by batteries or solar cell, which have limited energy. It is therefore important that these sensors operate optimally in order to enjoy a long service lifetime. Replacing batteries is not only very time consuming, but in some environments even not possible. The failure of the battery makes the final disappearance of the 'thing'.

Furthermore, wireless communication is inherently unreliable. When wireless sensor networks are connected to the Internet it is important that adequate quality can be offered because they are used for many different and often demanding applications. For some applications, the data must be received within a certain time, while for other applications the data must be received without errors. In addition, these different applications can be concurrently carried out over the same network of sensors, each with their specific requirements: a voice call that only may have a limited delay and a minimum of packet loss can be carried out simultaneously with an important data transfer which is not time-limited, but that needs a high reliability.

The premise of this thesis is to contribute to the success of the Internet of Things by consciously deal with energy and at the same time to offer several applications of sufficient quality.

In the first phase of this thesis, we focus on wireless sensor networks and the research challenges associated with the provisioning of sufficient quality guarantees.

After an introductory chapter, in Chapter 2, these research challenges are translated into design requirements and a general framework is presented that meets the requirements of different concurrent applications. We make a distinction between quality provision at the protocol level and at architectural level. In Chapter 3, the framework is on the one hand implemented and evaluated in an existing network simulator and on the other hand, the framework is integrated and validated in an existing sensor network architecture on real hardware in a testbed.

The second phase of this thesis elaborates on improving the energy consumption in sensor networks integrated with the Internet of Things'. A technique that is often used in wireless sensor networks to reduce energy consumption is in-network packet aggregation. When in-network aggregation is performed, many different small packets are encapsulated into one big packet. The idea behind in-network aggregation is that the fewer packets have to be sent, the less energy is consumed. This is mainly due to the fact that a sensor node consumes a lot of energy for wireless communication, i.e. for sending and receiving packets. A sensor at rest can go into a sleep mode in which hardly any energy is consumed by disabling the radio. Unfortunately, this often has a negative effect on the quality that can be offered. Aggregating means after all waiting on the right packets, and hence more delay. Therefore we study in Chapter 4 the impact of aggregation on energy consumption and on various quality parameters such as delay and reliability. Moreover, we investigate how aggregation can be performed within the IP standard used for the Internet of Things and within the QoS framework that was developed and implemented in Chapters 2 and 3.

In our study, we make a distinction between three aggregation scenarios: outgoing sensor network traffic that has to be routed towards one or more gateways with the Internet, local sensor network traffic and sensor network traffic that is coming from the Internet and has to be routed to many different sensor nodes.

In Chapter 5, we look at the outgoing traffic from the sensor network that has to be routed towards the Internet. Here we have mainly to do with the many-to-one relationship, where the traffic coming from multiple sensor nodes is routed to a gateway. There are often much aggregation capabilities, which can lead to substantial savings in terms of energy. More specifically, in this Chapter we are going to develop a mechanism which is able to dynamically adapt the degree of aggregation to the current application in order to always to be able to deliver the most optimal quality requirements. Hereby, there will always be a trade-off between energy consumption and the level of quality that is offered.

In Chapter 6, we analyze the traffic that is routed between the sensor nodes themselves. Here, there are often less aggregation capabilities, which is detrimental to the energy consumption. The traffic is often not routed towards one destination, but there are many connections between different sensor nodes. Where Chapter 5 focused on improving existing unicast aggregation techniques in which packets with the same destination are aggregated, we examine in this chapter the use of broadcast aggregation. With broadcast aggregation, instead of waiting on enough packets with the same destination to aggregate into one packet, packets are sequentially aggregated regardless of their destination. The resulting aggregated

packet is sent as a broadcast packet. The receiving nodes can then see whether this aggregated packet contains a packet that was destined for them and if so, then the corresponding packet can be filtered out of the aggregation packet.

Finally, we investigate in Chapter 7 the scenario whereby traffic from the Internet travels towards one or more sensor nodes of a sensor network. Again, standard unicast aggregation will not lead to optimal energy consumption since packets will almost always have different destinations. In this chapter we will again examine if broadcast aggregation may lead to better energy and QoS results.

In Chapter 8 the insights of this dissertation are summarized and some outlines for future research are turned out.

1 Introduction

1.1 Context

Wireless sensor networks are networks that contain a huge amount of small and cheap sensor nodes that are communicating through a low-power radio interface. They are typically used for large-scale sensing tasks and they are relaying the sensed information to a central base station, where the collected information is analyzed [1], as is shown in Fig. 1.1.

Traditional wireless sensor network applications are monitoring and tracking applications and process control [2]. In the past, each independent application used to have its own dedicated sensor network. Today, sensor networks are gaining popularity due to their plug-and-play nature and their low installation costs. As a consequence, wireless sensor networks are more and more interconnected with the Internet, referred to as the Internet of Things (IoT). For example, Libelium [3] lists 57 sensor applications for a smarter world situated in home, environments, industry, health care and many more. Sensor nodes join the Internet dynamically and use it to collaborate and accomplish their task [4]. However, wireless communication is unreliable by nature so providing Quality of Service in wireless sensor networks cannot be neglected. Furthermore, next-generation wireless sensor networks may simultaneously support diverse applications, each of them having its own specific QoS requirements, which can dynamically change in time and place. For instance, the same sensor network can be used for video surveillance, periodic smoke/fire monitoring and emergency support. Moreover, these applications can

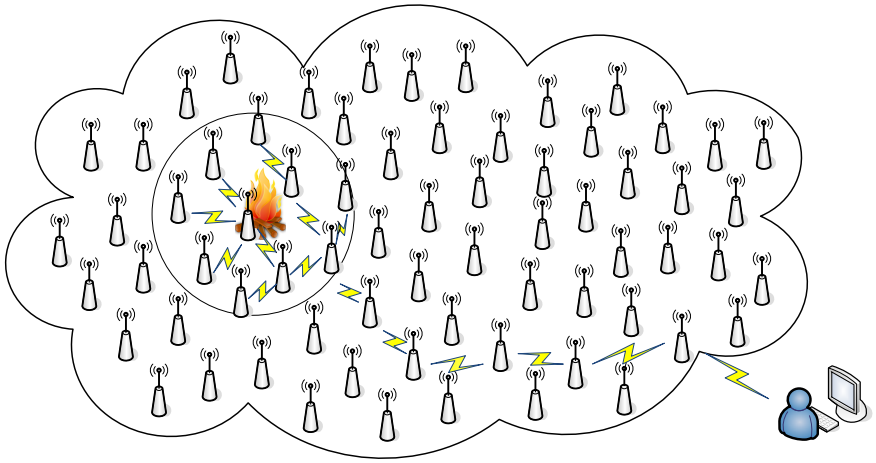


Figure 1.1. Traditional wireless sensor network

be applied across different network technologies and on top of different hardware.

Providing Quality of Service in terms of delay and reliability is not the only concern. Since a typical wireless sensor network is equipped with low-cost and low-power sensor nodes that are generally battery powered, frequent replacement of batteries should be avoided as much as possible, certainly when they are deployed in hard to reach locations. As a consequence, QoS should be supported with low energy consumption into mind.

Until now, most QoS efforts in wireless sensor networks are limited to the protocol level either through making network protocols QoS-aware [5, 6] or through cross-layer interactions [7]. The major shortcomings are that they mainly focus on a few QoS parameters such as reliability [8] and delay [9] while others such as throughput and energy are most of the time not considered. Additionally, these protocols are tuned to a specific network environment or a specific application. Finally, although the QoS and energy trade-off is very important, it is mainly ignored in current QoS research.

This PhD research aims to develop a general-purpose QoS solution for wireless sensor networks that are interconnected with the Internet and that is able to adapt itself to dynamic and time-varying application, network and node conditions, taking the QoS and energy trade-off into account. The challenges and shortcomings that will be addressed in this PhD research are introduced in more detail in the following sections.

1.2 Application Domains

Fig. 1.2 gives an overview of wireless sensor network application domains in which providing Quality of Service plays a mandatory role. These application domains are discussed below in more detail.

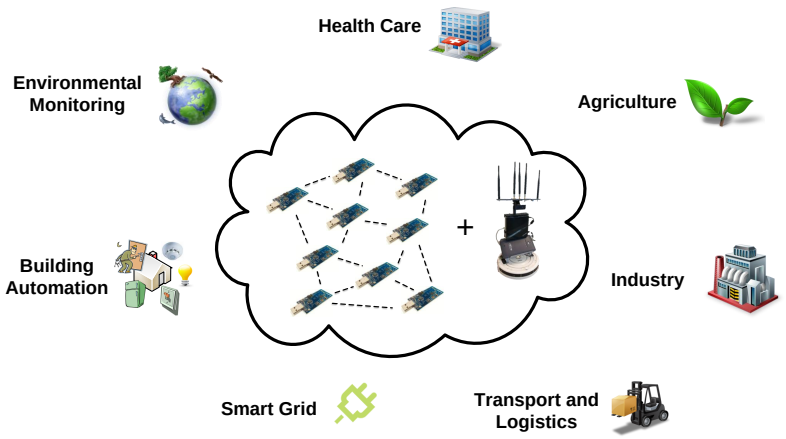


Figure 1.2. Mobile wireless sensor networks: use cases

1.2.1 Health Care

An example of a health care use case can be found in the iMinds DEUS project [10] in which elderly people and caregivers in nursing homes are monitored and tracked. Part of this PhD research has been performed in the context of the DEUS project.

In this project, academic partners, industry and non-profit organizations are collaborating on the development of a generic cost-efficient sensor network platform that can be easily deployed in diverse user scenarios. In one of the use cases, the sensor network has to assist elderly persons, in particular persons with dementia, in and around residential care homes. As elderly people often stray, they are kept in an isolated wing of the care home today.

The DEUS project has designed solutions offer elderly person more freedom and mobility. The DEUS solutions allow to deploy several services on top of a single sensor network: (i) indoor (within the care home) & outdoor (garden and

walking area around the home) positioning, (ii) emergency call realized through an alarm button on a sensor device carried by the elderly person, (iii) voice call between the elderly person and a nurse automatically established upon an emergency call or when the elderly person is located in an area where he/she is not allowed to go. These services have different requirements in terms of reliability, delay and bandwidth and the sensor network is expected to guarantee the different QoS requirements despite the dynamic nature of the wireless environments and the resource constraints (energy, bandwidth, processing capacity, memory) inherent to sensor networks.

1.2.2 Smart Grid and Building Automation

Due to their low cost and robust design, wireless sensor nodes are ideal for smart grid and building automation applications. Monitoring and signaling of smoke and fire, video surveillance, climate control, and smart grid monitoring and data collection are only a few of the many possible applications that need a certain QoS level in terms of reliability, throughput and delay. An example can be found in the Amsterdam Smart City project [11].

1.2.3 Transport and Logistics

Another application domain is transport and logistics. Wireless sensor nodes can be easily installed along traffic paths, in vehicles (cars, trucks, trailers) and on containers that should be tracked.

The MoCo (Monitoring Containers) project [12] aims to design a wireless network system that follows up products, stored in containers. During the transport of goods, it is important to have an accurate view on the condition of goods, on the trajectory and on the 'safe trade lane' conditions of the container. Since a continuous connectivity has to be guaranteed, QoS plays an important role.

In the NextGenITS [13] project aims to develop and demonstrate a number of Intelligent Transport Services (ITS) which have either or both a big social and commercial potential. Examples of QoS-aware services are: traffic information, intelligent speed adaptation (crash prevention), road charging (toll collection) and eCall (notification of the rescue services in case of an accident).

1.2.4 Industry, Agriculture and Environmental Monitoring

Even in industry, agriculture and environmental monitoring applications, at least a basic Quality of Service level is needed in order to guarantee a proper working of the deployed wireless sensor network. Examples are irrigation, animal population monitoring, detecting production process variations and prediction of maintenance.

In Australia, there exist several drinking water projects. One of them is the Wivenhoe Project that continuous monitors the quality of drinking water in the reservoir [14].

The authors of [15] have developed a sensor network for outdoor surveillance and monitoring of human living and working environments. In this case, robots are used to replace human presence in everyday working dirty, health-destructing, tiresome and monotonous jobs.

In [16], a chemical accident emergency search and rescue system was developed. It is used for monitoring the leakage of hazardous chemicals and guaranteeing the safety of people in such areas.

[17] combines wireless sensor networks and unmanned aerial vehicles (UAV) in isolated areas where communication between ground nodes might be difficult (for instance when coverage over large areas is necessary). The UAVs are then used to upload information sensed by sensor nodes on the ground or to deploy nodes. Where the UAVs can sense over a large area at high altitude, sensor networks can monitor over very small areas.

1.3 Definition and Metrics

Quality of Service (QoS) has many definitions, mostly depending on the field applied. Definitions range from objective to subjective measurements, and from concrete low-level to abstract high-level definitions. In this thesis, QoS is defined as an objective and explicit representation of the services delivered by the network, expressed in terms of delay, packet loss rate, packet delay variation, throughput and energy consumption. These QoS metrics are discussed below in more detail.

Delay End-to-end delay is the time it takes to transmit a packet from source to destination. This time includes queuing delays, propagation delays (related to the distance and medium between the nodes), transmission delays (caused to the data rate of the communication link) and processing delays. Most wireless sensor network applications will have some delay constraints. For real-time applications, these delay constraints can be very stringent.

Packet Loss Rate The packet loss rate is the amount of packets that may be lost during transmission and that is tolerated within the reliability requirements. Factors that can cause packet loss are congestion, broken communication links due to node failures or mobile nodes, and bit errors due to noise, interference, distortion or bit synchronization.

Packet Delay Variation Packet delay variation is the difference in end-to-end delay between sequential packets in a flow, ignoring any lost packets. This QoS

metric becomes important for interactive real-time applications, such as voice.

Throughput Throughput is defined as the average rate of successful delivered data from source to destination. The difference with bandwidth is that bandwidth defines the amount of data that physically can be transmitted through the medium (measured at the physical layer), while throughput is the actual data (measured at the application layer) that successfully reaches the end destination. Throughput hence also takes into account packet loss and delay.

Energy Consumption In wireless sensor networks, energy consumption can be defined as the amount of power that is used to perform wireless communication tasks and is expressed in Joule (J).

1.4 QoS Research Challenges

1.4.1 Dynamic Network Topology

Fig. 1.3 gives an overview of the four network topologies that are common in wireless sensor networks: point-to-point, star, tree and mesh. In a point-to-point network topology, each node can directly communicate with another node. In a star network on the other hand nodes are directly connected to a centralized node and can only communicate with each other via this centralized node. In a (cluster) tree network, there is one node on top, a sink or gateway node, which is the parent of the nodes on the level below (first level). These first level nodes are on their turn parent of the nodes one level lower (second level nodes). The total number of levels will depend on the size of the network. Finally, in a mesh network topology, each node is able to communicate with each other node, either directly or via multiple hops involving other nodes.

QoS has to be guaranteed independent of the applied network topology. Furthermore, where traditional wireless sensor networks mainly contain static sensor nodes, today's sensor networks also contain mobile nodes which may lead to faster changing network topologies compared to static sensor networks. QoS challenges in static sensor networks are often caused by unreliable links due to fading, shading due to obstacles, environmentally related link challenges (underwater/underground) and node fall out due to damage, battery depletion or displacement by nature (wind, explosion, ...). Unpredictable links due to mobility are an additional challenge in sensor networks with mobile nodes. In addition, mobile sensor networks often operate in more extreme environmental conditions such as earthquakes, chemical disaster environments and flooding.

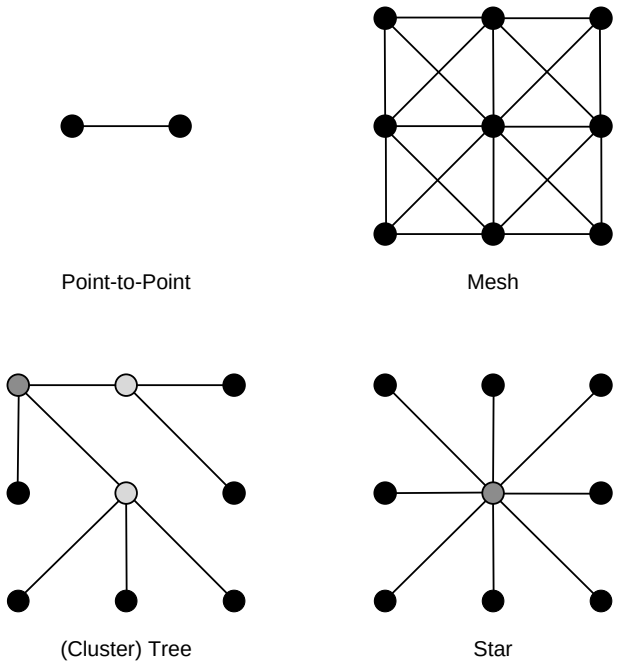


Figure 1.3. Sensor network topologies

1.4.2 Constraint Nodes

Sensor nodes are tiny low-cost and low-power devices with limited resources on processing, energy, communication range, bandwidth and memory. For instance, the Tmote Sky sensor node shown in Fig. 1.4 has a CC2420 radio with a theoretical bit rate of 250 kbps and an 8 MHz MSP430 microcontroller with only 10k RAM and 48k Flash memory. Table 1.1 compares the specifications of today's sensor nodes with more powerful nodes.

Due to the limited capabilities of sensor nodes, both the amount and the range of exchanged information are very limited and can be different for each kind of sensor node. Moreover, since energy is a scarce resource and sensor networks are often deployed in hard to reach environments that are complicating frequent battery replacements, energy saving mechanisms must be applied to extend the lifetime of the network. However, this may have a negative impact on the delivered QoS level. For example, applying sleeping schemes to save energy will lead to a higher end-to-end delay.

Mobile nodes on the other hand are high-end but expensive devices. They are generally equipped with GPS, a camera and computer ports. Although they also operate on batteries, they can be more easily recharged, as mobile nodes can

Specification	Tmote Sky	Zolertia Z1	Mobile Node
CPU type	TI MSP430 @ 8 MHz	2^{nd} gen. MSP430 @ 16 MHz	Intel Atom D510 @1,66 GHz Dual core
RAM	10 kB	8 kB	4 GB
Non-volatile memory	48 kB	92 kB	160 GB
Bandwidth	250 kbps	250 kbps	300 Mbps
Radio	CC2420	CC2420	Ateros AR9280
OS support	TinyOS	TinyOS Contiki	Windows Linux
Sensors	Temperature Humidity Light	Temperature Accelerometer	Temperature Humidity Light
Other			Bluetooth Webcam

Table 1.1. Hardware specifications of the different w-iLab.t nodes

Figure 1.4. Tmote Sky sensor node

automatically return to their base docking station after finishing their tasks. So there is no need for replacing batteries like in static sensor nodes.

These differences in storage, memory and processing create additional QoS challenges when applications are deployed over these heterogeneous nodes involving both low-cost static sensor nodes and high-end mobile nodes. For instance, it is difficult to route high-bandwidth applications partly over mobile nodes and partly over static nodes. Furthermore, where static sensor nodes mainly suffer from packet loss and delay issues due to the limited resources and the unreliable links, it is challenging to achieve low latencies due to frequent path changes.

1.4.3 Heterogeneous Networks

Wireless sensor networks can be deployed on top of different communication technologies and network protocols. For terrestrial sensor networks, IEEE 802.15.4 is generally used, while between mobile nodes, IEEE 802.11 is more commonly used. Other communication technologies can be applied for networks deployed underwater, underground or aerial. There may be a need for information exchange

between these different networks. For instance, underground or underwater collected information can be transported over a terrestrial sensor network. Underwater networks are characterized by acoustic communication with very limited bandwidth, very long delays and a very high bit error rate [18]. Underground networks suffer from extreme path losses and low data rates [19]. These challenges should be taken into account when developing a QoS solution. Such a solution should not only work for each network individually, but also across different networks.

1.4.4 Heterogeneous Applications

Wireless sensor networks are used for diverse applications in many domains. Applications not only differ from sensor network to sensor network, but they can also evolve in time or different applications can concurrently run on the same network. Each application will have its own QoS requirements on delay, packet loss rate, throughput and packet delay variation. For instance, a sensor network can be used for simultaneous reliable data collection and delay-sensitive voice and video monitoring. As a consequence, the network should be able to continuously adapt itself to the instantaneous application requirements.

Wireless sensor network applications often have one of the following communication types: point-to-point (P2P), point-to-multipoint (P2M), multipoint-to-point (M2P) and multipoint-to-multipoint (M2M). These communication types are illustrated in Fig. 1.5. In point-to-point communication, there is a dedicated communication path between two nodes, while in point-to-multipoint communication there is a common communication path from a single source node to multiple destination nodes. Such a common communication path could be realized in different ways: (1) either by establishing multiple dedicated paths from the source to multiple destinations, (2) or by setting up a multicast tree hereby avoiding duplicate transmissions of traffic, (3) or through broadcasting the traffic to all nodes. In multipoint-to-point communication, one destination node can receive messages from different source nodes while in multipoint-to-multipoint communication each node can send messages to and receive messages from all other nodes.

Table 1.2 illustrates for each sensor network application domain from Section 1.2 an example application with its different QoS metrics and their communication type.

1.5 Outline

In Section 1.4, the problems and challenges for providing QoS in wireless sensor networks have are formulated that are tackled in the remainder of this PhD dissertation. In this section we present the outline of this dissertation and explain how the different chapters are linked together. A schematic positioning of the differ-

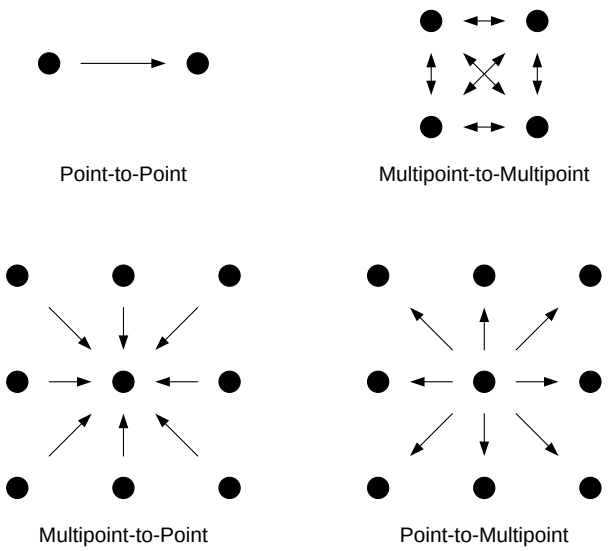


Figure 1.5. Sensor network communication types

ent research contributions that are further detailed in Section 1.6, can be found in Fig. 1.6. The complete list of publications that resulted from this work is presented in Section 1.7

In Chapter 2, the QoS challenges are translated in design requirements. The QoS Framework is designed with these requirements in mind. The focus is on the trade-off between energy consumption and QoS requirements. To this end, the QoS Framework is divided in two parts: a protocol-independent and a protocol-dependent QoS support part. While the former implements basic QoS functionalities, the latter implements more advanced (and often more energy consuming) QoS functionalities. In addition, the modular design easily allows to implement only parts of the modules, which can differentiate between nodes with different capabilities.

Implementation, evaluation and validation of the QoS Framework is done in Chapter 3. The QoS Framework is validated both in a simulator and on real hardware. Simulations were executed using the Castalia wireless sensor network simulator, while the implementation on real hardware is done on TmoteSky sensor nodes running the TinyOS operating system and the nesC programming language.

Since the trade-off between energy consumption and QoS requirements is very important, Chapter 4 focuses on how in-network aggregation can reduce energy consumption. In-network aggregation is a technique in which several small packets are aggregated into one bigger packet in order to limit the number of transmissions. Since most sensor network energy is consumed by the radio for sending

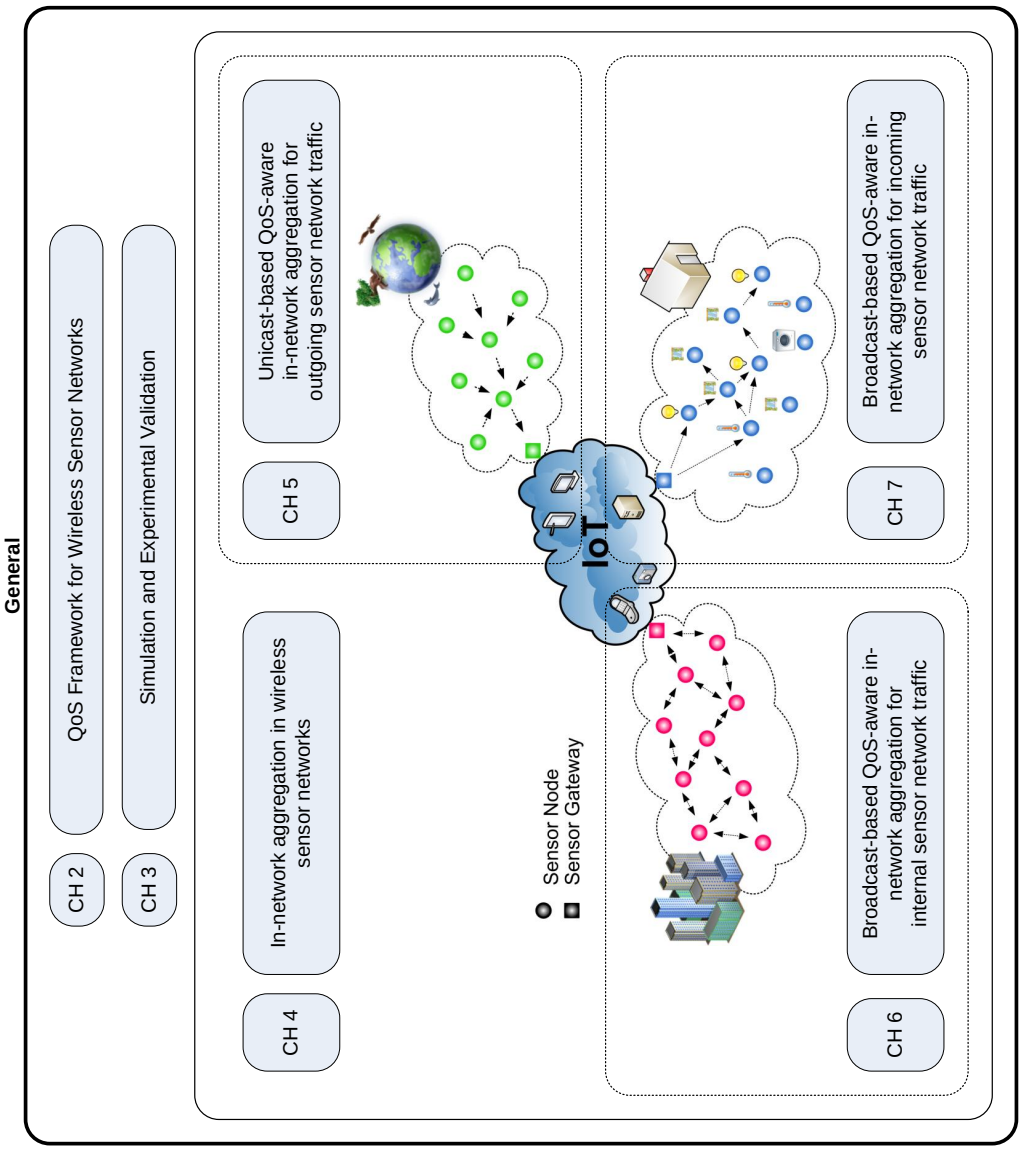


Figure 1.6. Schematic position of the different chapters in this dissertation

Application	**Comm. type**	**Delay (in s)**	**BER (in %)**	**PDV (in ms)**	**TP (in kbps)**	**Energy**
eHealth emergency call	P2P	0.100	$< 10^{-2}$	400	60-80	medium
video surveillance	P2P/ M2P	0.150	$< 10^{-3}$	100	28.8-500	high
tracking information	P2P/ P2M	5	0	-	-	low
environmental temperature monitoring	M2P	2	> 1	-	-	low

Table 1.2. Specifications of several sensor network applications together with their Communication Type (Comm. type) and QoS metrics: Delay, Bit Error Rate (BER), Packet Delay Variation (PDV), Throughput (TP) and Energy

and receiving data packets, reducing the number of transmissions will reduce the total energy consumption and as such extend the sensor network lifetime. In this thesis, the focus is on scenario's in which sensor networks are interconnected with the Internet of Things. This interconnection results in three traffic paths: outgoing sensor network traffic that has to be routed towards one or more gateways, local sensor network traffic and incoming sensor network traffic that has to be routed to many different sensor nodes.

The outgoing sensor network traffic is addressed in Chapter 5. The impact of in-network aggregation on energy and on QoS parameters such as delay and reliability is analyzed. Furthermore, a tunable protocol-independent QoS-aware in-network aggregation scheme that is able to make a trade-off between energy consumption and the requested QoS level is developed. Results show that the unicast-based in-network aggregation scheme works well, as long as there are not too many gateways. The unicast-based is hence not suited when there are many different source and destination sensor nodes. Therefore, Chapter 6 shows how broadcast-based in-network aggregation can increase the aggregation possibilities for local sensor network traffic where there exist many communication paths between the different sensor nodes. Finally, Chapter 7 tackles the incoming sensor network traffic from the Internet towards many different sensor nodes and investigates the possibility to use broadcast aggregation for this traffic pattern.

In Chapter 8 the insights of this dissertation are summarized and some outlines for future research are turned out.

1.6 Research Contributions

The main research question of this project was: "How to provide sufficient QoS guarantees for demanding applications in wireless sensor networks that are interconnected with the Internet of Things and that have limited resources in terms of energy, bandwidth, processing and memory capacity". This research question can be investigated on two different levels, which leads to two more detailed research questions:

- How can QoS be provided at an architectural level? Are there already options to cope with QoS as part of an (existing) sensor network architecture without the need to adapt all network protocols in the protocol stack?
- How can protocols be optimized to meet the requested QoS level?

These research questions lead to following contributions which are presented in this dissertation:

- Contribution 1: Development of a QoS framework for next-generation wireless sensor networks.
  We have analyzed the different QoS and energy needs that result from integrating wireless sensor networks and Internet of Things applications. This information is used to design a novel QoS framework that is able to adjust itself to the different constraints at node and network level, and to applications requirements. The framework is designed to allow implementation of protocol-independent as well as protocol-dependent mechanisms to meet the requested QoS level. Its modular approach allows flexibility and expandability. Several mechanisms are presented and their impact on the QoS level is illustrated. This QoS framework has been implemented and validated both in a simulation environment and in a real-life testbed using real hardware. Furthermore, this QoS framework is implemented in simulation and an experimental validation is made on real hardware.
- Contribution 2: Development of QoS-aware in-network aggregation protocol approaches for next-generation wireless sensor networks.
  We have investigated the impact of in-network aggregation on the provided QoS level for different traffic scenarios, and we have presented several mechanisms to improve the QoS level when in-network aggregation is applied. First, we have designed a unicast-based QoS-aware in-network aggregation approach for outgoing WSN traffic and afterwards, we have investigated to use of broadcast-based QoS-aware in-network aggregation for local WSN traffic. Finally, we have explored the use of QoS-aware broadcast in-network aggregation for incoming WSN traffic.

1.7 Publications

The research results obtained during this PhD research have been published in scientific journals and presented at a series of international conferences. The following list provides an overview of the publications during my PhD research.

1.7.1 Publications in international journals (listed in the Science Citation Index [1])

1. Eli De Poorter, **Evy Troubleyn**, Ingrid Moerman and Piet Demeester. *IDRA: a flexible system architecture for next generation wireless sensor networks.* Published in Wireless Networks, 17:1423–1440, 2011.

2. **Evy Troubleyn**, Ingrid Moerman and Piet Demeester. *QoS Challenges in Wireless Sensor Networked Robotics.* Published in Wireless Personal Communications, 70(3):1059–1075, June 2013.

3. **Evy Troubleyn**, Jeroen Hoebeke, Ingrid Moerman and Piet Demeester. *Broadcast Aggregation to Improve Quality of Service in Wireless Sensor Networks.* Published in International Journal of Distributed Sensor Networks, 1014(2014): 9 March 2014.

4. **Evy Troubleyn**, Ingrid Moerman and Piet Demeester. *Protocol-Independent QoS-aware In-Network Aggregation in Wireless Sensor Networks.* Submitted to Wireless Networks, 2014.

1.7.2 Publications in other international conferences

1. **Evy Troubleyn**, Eli De Poorter, Ingrid Moerman and Piet Demeester. *AMoQoSA: Adaptive Modular QoS Architecture for Wireless Sensor Networks.* Published in Proceedings of the Second International Conference on Sensor Technologies and Applications (SENSORCOMM), pages 172–178, Cap Esterel, France, August 2008

2. **Evy Troubleyn**, Peter Ruckebush, Eli De Poorter, Ingrid Moerman and Piet Demeester. *Supporting Protocol-Independent Adaptive QoS in Wireless Sensor Networks.* Published in Proceedings of the IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC), pages 253–260, Hyatt Regency Newport Beach, Newport Beach, California, USA, June 2010

[1] The publications listed are recognized as 'A1 publications', according to the following definition used by Ghent University: A1 publications are articles listed in the Science Citation Index, the Social Science Citation Index or the Arts and Humanities Citation Index of the ISI Web of Science, restricted to contributions listed as article, review, letter, note or proceedings paper.

3. **Evy Troubleyn**, Ingrid Moerman and Piet Demeester. *QoS Challenges in Wireless Sensor Networked Robotics.* Published in Proceedings of the 14th Strategic Workshop 2012: Wireless Robotics: Research and Standardisation, Abstracts, pages 1–21, Aalborg, Denmark, 2012

4. **Evy Troubleyn**, Lieven Tytgat, Ingrid Moerman and Piet Demeester. *Poster Abstract: Improving In-Network Aggregation by Exploiting the Broadcast Nature of Wireless Communication.* Published in Proceedings of the 10th European Conference on Wireless Sensor Networks (EWSN), pages 53–55, Ghent, February 2013

1.7.3 Publications in national conferences

1. **Evy Troubleyn**, Ingrid Moerman, Tom Dhaene and Piet Demeester. *Quality of Service in Heterogeneous Wireless Sensor Networks.* Published in the 10th UGent - FirW PhD symposium, pages 104-105, Ghent, Belgium, December 2009.

References

[1] I. F. Akyildiz, T. Melodia, and K. R. Chowdhury. *A survey on wireless multimedia sensor networks*. Computer Networks, 51(4):921–960, 2007. doi:10.1016/j.comnet.2006.10.002.

[2] J. Yick, B. Mukherjee, and D. Ghosal. *Wireless sensor network survey*. Computer Networks, 52(12):2292–2330, 2008.

[3] Libelium. *Top 50 IoT Sensor Applications Ranking*. http://www.libelium.com/top_50_iot_sensor_applications_ranking/. Online; accessed 3 April 2013.

[4] D. Christin, A. Reinhardt, P. S. Mogre, and R. Steinmetz. *Wireless sensor networks and the internet of things: selected challenges*. In Proceedings of the 8th GI/ITG KuVS Fachgespräch "Sensor Networks", pages 31–34, August 2009.

[5] M. Younis, K. Akkaya, M. Eltoweissy, and A. Wadaa. *On handling qoS traffic in wireless sensor networks*. In Proceedings of the 37th Annual Hawaii International Conference on System Sciences, Janari 2004. doi:10.1109/HICSS.2004.1265688.

[6] D. Chen and P. K. Varshney. *QoS Support in Wireless Sensor Networks: A Survey*. In Proceedings of the International Conference on Wireless Networks, pages 227–233, 2004.

[7] W.-Y. Cai and H.-B. Yang. *Cross-layer QoS optimization design for wireless sensor networks*. In IET Conference on Wireless, Mobile and Sensor Networks, pages 249–252, 2007.

[8] E. Felemban, C.-G. Lee, and E. Ekici. *MMSPEED: Multipath Multi-SPEED Protocol for QoS Guarantee of Reliability and Timeliness in Wireless Sensor Networks*. IEEE Transactions on Mobile Computing, 5(6):738–754, 2006. doi:10.1109/TMC.2006.79.

[9] M. Caccamo, L. Y. Zhang, L. Sha, and G. Buttazzo. *An Implicit Prioritized Access Protocol for Wireless Sensor Networks*. In Proceedings of the 23rd IEEE Real-Time Systems Symposium, page 39, 2002.

[10] S. Bouckaert, E. De Poorter, B. Latré, J. Hoebeke, I. Moerman, and P. Demeester. *Strategies and challenges for interconnecting wireless mesh and wireless sensor networks*. Wireless Personal Communications, 53(3):443–463, 2010.

[11] *Amsterdam Smart City*. http://www.http://amsterdamsmartcity.com/. [Online; accessed 17 March 2014].

[12] *MoCo (Monitoring Containers*. http://www.iminds.be/en/research/overview-projects/p/detail/moco-2. [Online; accessed 17 March 2014].

[13] *NextGenITS*. http://www.iminds.be/en/research/overview-projects/p/detail/nextgenits. [Online; accessed 17 March 2014].

[14] A. G. M Dunbabin, J Udy and M. Bruenig. *Continous monitoring of reservoir water quality: the Wivenhoe Project*. Journal of the Australian Water Association, 36(6), 2009.

[15] A. Rodic, D. Katie, and G. Mester. *Ambient intelligent robot-sensor networks for environmental surveillance and remote sensing*. In Proceedings of the 7th International Symposium on Intelligent Systems and Informatics, pages 39–44, September 2009. doi:10.1109/SISY.2009.5291140.

[16] H. Wang, M. Zhang, J. Wang, and M. Huang. *Engineering an emergency search and rescue application with wireless sensor network and mobile robot*. In Proceedings of the 2010 International Conference on Measuring Technology and Mechatronics Automation, volume 2, pages 112–115, March 2010. doi:10.1109/ICMTMA.2010.227.

[17] S. Teh, L. Mejias, P. Corke, and W. Hu. *Experiments in integrating autonomous uninhabited aerial vehicles (UAVs) and wireless sensor networks*. In Proceedings of the 2008 Australasian Conference on Robotics and Automation, March 2008.

[18] I. F. Akyildiz, D. Pompili, and T. Melodia. *Underwater acoustic sensor networks: research challenges*. Ad Hoc Networks, 3:257–279, 2005.

[19] I. F. Akyildiz and E. P. Stuntebeck. *Wireless underground sensor networks: research challenges*. Ad Hoc Networks, pages 669–686, 2006.

2

QoS Framework for Wireless Sensor Networks

2.1 Introduction

In Chapter 1, we showed that wireless sensor networks interconnected with the Internet of Things are used for many diverse applications. We also gave an overview of some applications together with their different challenges in terms of Quality of Service (QoS). To cope with these challenges in the next-generation wireless sensor networks, we present in this chapter our QoS Framework for the next-generation wireless sensor networks. This framework is able to adapt itself to dynamic and time-varying application, network and node conditions, taking the QoS and energy trade-off into account. To be able to design this framework, we start with translating the QoS challenges into design requirements in Section 2.2. Next, the actual QoS Framework design is presented in Section 2.3. Sections 2.4 and 2.5 discuss the protocol-independent and protocol-dependent QoS support in this framework respectively. In Section 2.6 we elaborate on the integration possibilities of this QoS Framework. Finally, we end this chapter with a short conclusion in Section 2.7.

2.2 Design Requirements

In this section, we elaborate on the QoS Framework design requirements that could be extracted from the different QoS challenges in today's wireless sensor network applications.

Adaptive The QoS Framework should be adaptive both in time and space. It should adapt itself in time to changing applications and dynamic environments due to mobile nodes or link breaks. It should also be adaptive in space because we aim at a generic QoS Framework that can be applied in different environments and networks (terrestrial, underwater, underground and aerial).

Energy-Efficient Since sensor nodes are often battery powered, replacing batteries is very time-consuming and to be avoided (especially in hard to reach environments). Energy is hence a scarce resource in wireless sensor networks. As a consequence, the QoS Framework should be designed with energy efficiency in mind, as the support of QoS tends to consume more energy. For instance, a lower delay will lead to shorter sleeping schemes and hence more energy consumption.

Scalable Sensor networks may contain thousands of sensor nodes. Therefore, the complexity of the designed framework should not grow considerably with the number of nodes.

Distributed Approach In a centralized network approach, a central control unit collects information, processes it and broadcasts the decisions or actions to the other (mobile) nodes. This behavior is not recommended in networks with thousands of nodes. Therefore, a distributed approach is desirable. Each node can take decisions locally and independently. However, a hybrid approach could also be beneficial, in which nodes with more capabilities can fulfill a centralized role.

Support Heterogeneity The QoS Framework should allow (1) different applications and should work on (2) nodes either applying existing (standardized) or new protocols. Furthermore, it should work for (3) each communication technology and (4) for nodes with different capabilities in terms of memory, processing and energy capacities. A lightweight framework can be used on nodes with low capabilities, while a more advanced framework implementation can be applied on nodes with more capabilities.

2.3 QoS Framework Design

Fig. 2.1 shows the design of the QoS Framework as part of a modular sensor network architecture. On top of this architecture, a Network Application Aggregator is responsible for managing the different applications and for access control. Below, the modular network architecture is shown. It contains the QoS Features, the Information Database and the Core System. The QoS Features contains a QoS Packet Policies, a QoS Protocol Policies and a QoS Management part. The Core System contains a Common Queue and the Network Protocols. Our QoS approach can be divided in a protocol-independent and a protocol-dependent QoS support part. This will be explained in more detail in the following sections.

2.4 Protocol-Independent QoS Support

In this section, we discuss how QoS can be supported in a protocol-independent way. The idea behind this protocol-independent approach is to provide QoS at an architectural level and thus extracting the QoS functionalities from the traditional layers/network protocols [1]. In a traditional approach, QoS needs to be supported at each network layer and each network layer has its own storage queue. This approach leads to duplicate functionalities, a narrow per-layer view and a waste of scarce energy [2]. Introducing a separate information database and working on a common queue has the advantage that the QoS system has a global network overview and that there is a load-balanced storage. A global network overview is beneficial when packets need to be dropped, or when the most suited packets for processing or sending should be selected. The load-balanced storage provides an overall optimal queue size, because high storage requirements for one layer can be compensated by a lower storage needed for another layer.

Fig. 2.2 shows this protocol-independent QoS design. The main components are the Information Database, the Common Queue, the QoS Packet Policies and the QoS Management and are discussed below in more detail.

2.4.1 Information Database

The Information Database is responsible for storing metadata. For the protocol-independent QoS support part of the framework, the part of the applications is the most important one. Applications can register QoS-specific information on delay, packet loss rate, packet delay variation and throughput. Some examples are given in Table 2.1.

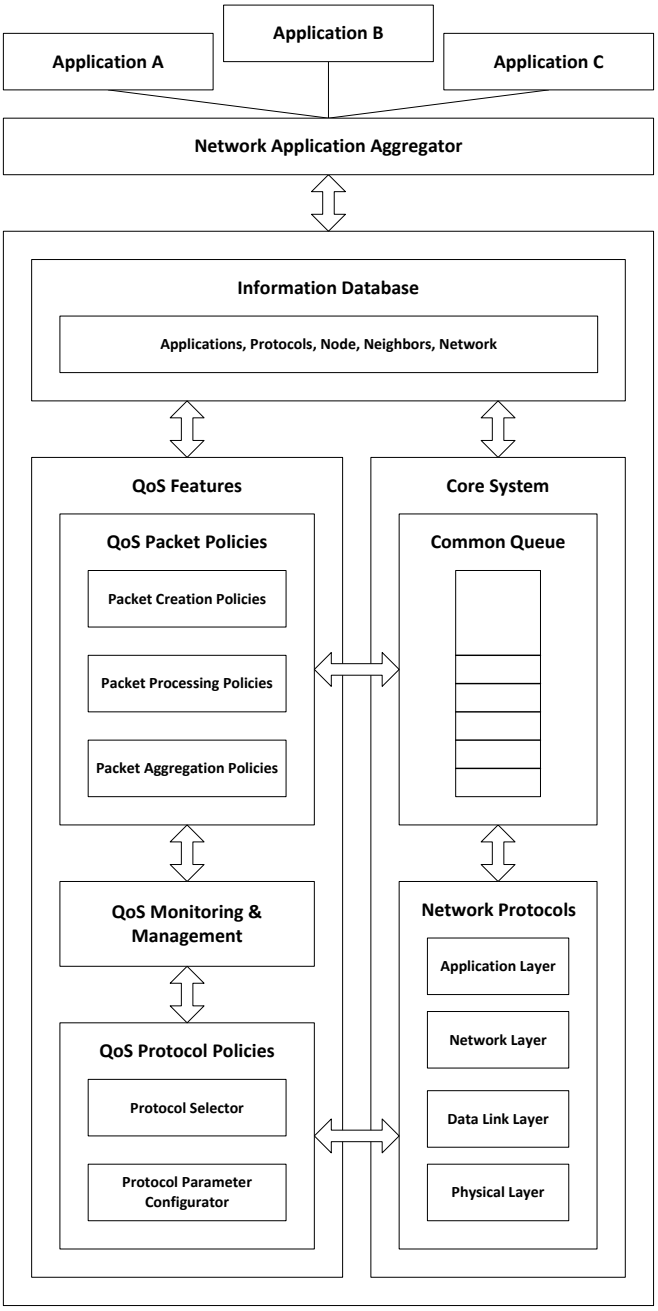


Figure 2.1. General QoS Framework

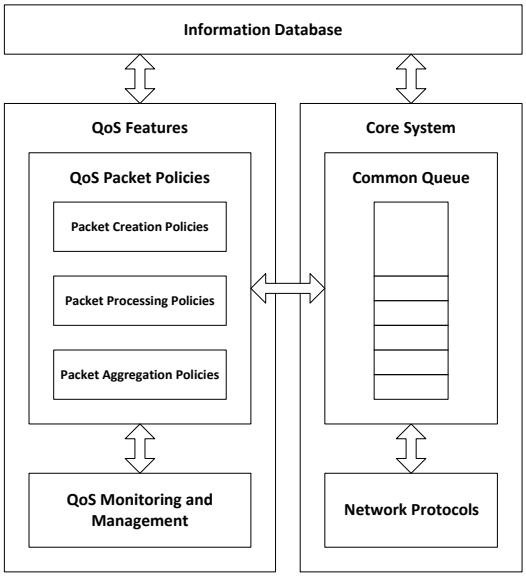


Figure 2.2. Protocol-independent QoS support

Network Protocol	Parameter Name	Metadata Name	Metadata Value
Monitoring Application	temperature	packet loss rate	20%
Voice Application	voice	end-to-end delay	200 ms

Table 2.1. Information Database

2.4.2 Common Queue

The Common Queue is responsible for storing the data packets. These can be incoming packets, locally created packets or packets that have to be forwarded. These packets are waiting in the Common Queue until they can be processed by the responsible protocol. On this Common Queue, several QoS mechanisms are applied, which are discussed below.

2.4.3 QoS Packet Policies

QoS Packet Creation Policies The QoS Packet Creation Policies are responsible for adding a QoS header to the data packets. This QoS header combines a mandatory QoS priority level and one or more optional QoS attributes, as shown in Fig. 2.3. The header lasts between 3 bits and up to 8 bytes, depending on the QoS attributes added (see below).

Handling heterogeneous applications in mobile wireless sensor networks re-

Figure 2.3. QoS Header

quires handling different traffic flows. Two QoS-oriented technologies that are common for traffic differentiation in IP networks are IntServ [3] and DiffServ [4]. IntServ is a flow-based approach which treats each flow individually. This approach is very flexible, but since each node has to maintain per flow state information, it is not scalable. The approach is also often too complex to use on sensor nodes with limited capabilities. DiffServ on the other hand is a class-based approach which differentiates between services by introducing some service classes using the DSCP field. The advantage is that complex operations move to the edge routers, while the core routers remain simple. The drawback is that quantitative information of each flow is lost after aggregation in service classes.

Our approach combines the best of both approaches:

- The simplicity and scalability of a class-based approach
- The flexibility of a flow-based approach

To realize this, a fixed number of QoS priority levels are defined. To add more flexibility, optional QoS attributes can be added to each packet. These attributes fulfill fine-grained packet control within the limits of the chosen priority level.

In Table 2.2, a proposal for these QoS priority levels is given. The three highest priority levels are reserved for MAC, Routing, and Monitoring and Management control information. This avoids deadlock situations, for instance, it prevents that packets cannot be sent because the MAC protocol is not up and running. Furthermore, there is a distinction between real-time traffic, time-sensitive traffic and best effort traffic. Within the real-time traffic and time-sensitive traffic classes, a small distinction can be made between critical and default. This can be justified by the following example. Suppose two simultaneous voice calls are traveling through the network. One voice call is close to the maximum delay, while the other has still time left in order to arrive on time. The first call can then be given a higher priority in order to allow this call to arrive on time.

To differentiate between these QoS priority levels and make the approach flexible, additional QoS attributes can be added to each packet. Examples of these QoS attributes are given in Table 2.3. When the application is not only time-sensitive, but also requires reliable data transport, a reliability-attribute can be added. The network protocols will then use these attributes to the best of their abilities. For instance, the MAC module can request an ACK message to ensure reliable transmission, and the QoS Framework can decide to drop a packet with a low reliability

QoS Priority Level	Description
7	Reserved (MAC control information)
6	Reserved (Routing control information)
5	Reserved (Monitoring/Management information)
4	Real-time traffic (critical mode)
3	Real-time traffic (default mode)
2	Time-sensitive traffic (critical mode)
1	Time-sensitive traffic (default mode)
0	Best Effort traffic

Table 2.2. QoS Priority Levels

indication when the Common Queue is full.

Attribute	Description	Size
Current_Delay	Traveled packet delay until now	2 bytes
Max_Delay	Max. allowed end-to-end packet delay	2 bytes
Reliability	Packet reliability indication	3 bits

Table 2.3. QoS Packet Attributes

Note that the QoS Framework is responsible for setting the priority levels and attributes. The application or network protocol developer can only register its requirements in the Information Database.

QoS Packet Processing Policies The QoS Packet Processing Policies are responsible for diverse processing rules. Firstly, they contain the packet selection rules. These rules define when a packet should be selected for processing and sending. This decision can be based on the priority level only, or also on QoS attributes such as delay and reliability when available. Secondly, the packet drop rules are defined. They define when a packet should be dropped and which packet should be dropped. For example, when the queue is completely full, it can drop the packet with the lowest reliability or a packet which deadline has almost been reached and, as a consequence, has a high drop probability in one of the following nodes.

QoS Packet Aggregation Policies As already stated, energy is a scarce resource in sensor networks. Therefore, the QoS Framework has to implement a mechanism to reduce the consumed energy. Because in-network aggregation is a well known technique in sensor networks to reduce energy, the information aggregation policies defines when aggregation has to be performed. This aggregation is based on a

trade-off between energy requirements and QoS requirements. More aggregation leads to a reduced energy consumption, but also increases the end-to-end delay.

These aggregation rules also define what should happen if packets with different QoS headers have to be aggregated. They can have a different priority level, but also different QoS attributes.

The Packet Creation, Packet Processing and Packet Aggregation Policies can be predefined (by performing a design time exploration study, from which we can determine the influence of decision parameters of the wireless system) or they can be defined at runtime through learning strategies, such as reinforcement learning [5].

2.4.4 QoS Monitoring and Management

The QoS Monitoring and Management module is responsible for monitoring the information in the Information Database and, based on this information, defining/updating the Packet Creation, Packet Processing and Packet Aggregation Policies. For example, when the remaining energy of the node becomes low, less energy consuming aggregation rules can be selected. Another example is that the priority of the packets can be changed with changing application requirements. This way, a voice application with very stringent delay requirements can receive a (temporarily) higher priority level (from default mode to critical mode). It is also possible that packets needing a high-bandwidth communication path have to be processed first in case a mobile node that can guarantee a reliable communication path to the sink comes within the reach of the node.

2.4.5 Discussions

The Information Database in which several application requirements can be stored, enables an *adaptive* approach. QoS settings will be based on this information, and changing requirements in the Information Database lead to changing QoS settings. The QoS header defined in the QoS Packet Creation Policies and the Packet Processing Policies leads in turn to *scalability* and *application heterogeneity* because each application can define its own specific requirements. Because this QoS header is sent together with the packet, a *distributed* approach is possible. Each node can take decisions based on this information and fulfill the packet's requirements to the best of its abilities. *Energy-efficiency* is met by adding QoS Packet Aggregation Policies. Because each node can implement (part of) the functionalities based on their capabilities, *node heterogeneity* is reached.

2.5 Protocol-Dependent QoS Support

The protocol-dependent QoS support part is displayed in Fig. 2.4. The main components of this part are the Information Database, the QoS Management, the QoS Protocol Policies and the Network Protocols.

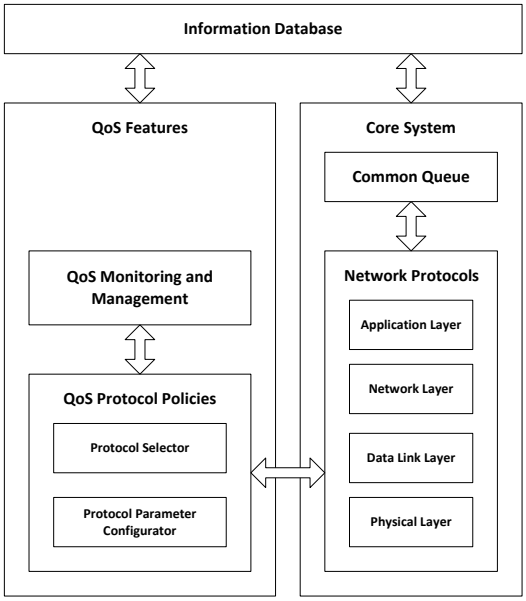


Figure 2.4. Protocol-dependent QoS support

While the protocol-independent part decouples the QoS support from the network protocols, the protocol-dependent part interacts with the available (QoS-aware) network protocols.

2.5.1 Information Database

The Information Database has the same functionality as in Section 2.4.1, namely storing metadata. For protocol-dependent QoS support, the information on protocols, nodes, neighbors and the network is used. For instance, the nodes and neighbors information database can specify if a node or neighbor is a mobile robot or a simple static sensor node, it can contain the amount and the position of neighboring mobile nodes and it can contain an indication of the remaining energy. This information can be used by the QoS Protocol Policies (see Section 2.5.2) when taking routing decisions. For example, a Collection Tree Protocol [6] that relays information to the sink will avoid mobile nodes that are passing by, as such nodes have fewer stable links. On the other hand, for high-bandwidth data, the high-capacity

mobile nodes will be preferred for relaying this information. Furthermore, protocols can register information on the maximum QoS requirements that they can deliver (high reliability, low delay, ...) and information on the network density and the overall network performance can be made available.

2.5.2 QoS Protocol Policies

The QoS Protocol Policies contains two functionalities: a Protocol Selector and a Protocol Parameter Configurator. These tools allow an optimal tuning of the network to the instantaneous QoS requirements. The Protocol Selector can select the most appropriate network protocol, for instance the DYMO [7] routing protocol for routing point-to-point voice traffic or a CTP routing protocol for routing monitoring information from different source nodes to one sink node. From the routing table (in case multiple routing protocols are available, this will be a shared routing table) and the neighborhood table, the Protocol Selector can select the most stable link or best routing path. This could be a direct neighborhood-link or a multi-hop routing path. Based on this information, the most optimal routing protocol can be selected.

The Protocol Parameter Configurator is responsible for an optimal tuning of the protocol parameters, for example, the optimal route timeout can be set for a routing protocol, or the MAC slots or the sleeping schedule can be tuned in an optimal way for a voice routing call.

As in the protocol-independent part, both the Protocol Selector and the Protocol Parameter Configurator can take decisions based either on pre-defined rules or on runtime learning strategies.

2.5.3 QoS Monitoring and Management

Comparable to Section 2.4.4, the QoS Monitoring and Management module is responsible for monitoring the information in the Information Database and defining/updating both the Protocol Selector and Protocol Parameter Configurator Policies. For instance, based on the load, the remaining energy of the network and the existence of mobile nodes, the most optimal routing and MAC protocols can be selected. The protocol parameters can be tuned to the instantaneous network condition or application requirements. For example, the MAC sleeping time can be tuned to the remaining energy level and to the requested QoS level.

2.5.4 Discussions

The possibility to tune and replace network protocols at runtime ensures *adaptivity* to changing environments and applications. Since an independent group of nodes in a certain area can decide to tune or replace the parameters, the approach

is *distributed*. Protocol parameters can be tuned to the instantaneous optimal settings which leads to high *energy savings*. *Network heterogeneity* is reached since protocols can be tuned and replaced based on the network and communication technology.

2.6 QoS Framework Integration

The developed QoS Framework can be implemented in any existing (layered) network architecture, but due to its modularity, it is optimally suited for a cross-layer or layerless approach.

According to the OSI Reference Model [8], supporting QoS is one of the few network functionalities that is involved in all system layers. Advantages of supporting QoS in such a layered structure are its standardized interface and its transparency. Many studies [2] argue that the layered protocol architecture, where each layer is designed and operated independently and which works very well for wired networks, seems to be very inefficient when deployed in wireless networks, which are much more dynamic and less predictable. Furthermore, this layered approach requires each layer to define its own QoS header information which gives much overhead on limited sensor nodes. Further, the QoS capabilities of each layer are restricted by the QoS capabilities of the layer above and below.

Many cross-layer design ideas, going from the direct communication between layers up to complete layerless architectures, have already been presented [9, 10]. Direct communication between layers is the most straightforward way for cross-layer interaction and makes variables at one layer visible to the other (non-adjacent) layers at run-time. Another approach is based on a shared database, whereby the database can be regarded as a new layer, providing the service of storage/retrieval of information to all layers. The main challenges are to adapt the single layer protocols taking into account the information obtained from other layers and to design interfaces either between layers or between layers and a shared database. Yet another approach is joint optimization between adjacent layers. The disadvantage of such an approach is however its unclear structure and the inherent difficulty to deal with new layers in a plug and play manner.

In a layerless architecture, the protocols are organized in a modular way, enabling plenty of interactions between the different modules. While the latter approach offers the greatest flexibility, it is not anymore compatible with standard layered protocol stacks as is the case for the former cross-layer approaches. Layerless architectures may however be very effective for sensor networks, where devices have constrained memory, processing and battery capabilities.

2.7 Conclusion

Next-generation mobile wireless sensor networks used as interconnection with the Internet of Things is a booming research area. They are used for monitoring environments, health care, home and building automation, tracking, and even in consumer products such as house cleaning and gardening. This area of possible applications creates a wide range of requirements on the delivered Quality of Service level.

In order to cope with these QoS requirements, we have presented a general QoS Framework that can be integrated in any existing or future network architecture. A basic QoS level can be guaranteed with a protocol-independent approach. It works independent of the available network protocols. For a more in-depth QoS level, the protocol-dependent approach allows tuning of protocol parameters and replacing network protocols in a distributed manner. This way, the network can adapt itself to the instantaneous best QoS and network behavior.

A big advantage of our QoS Framework is that due to its modular approach, it is optimally suited to be integrated in any future cross-layer or layerless wireless sensor network architecture. Such architectures have already proven to be more efficient for wireless (sensor network) design.

References

[1] E. Troubleyn, E. De Poorter, P. Ruckebusch, I. Moerman, and P. Demeester. *Supporting protocol-independent adaptive qoS in wireless sensor networks*. In Proceedings of the 2010 IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC)., pages 253–260, June 2010. doi:10.1109/SUTC.2010.34.

[2] V. Srivastava and M. Motani. *Cross-layer design: a survey and the road ahead*. Communications Magazine, IEEE, 43:112–119, 2005.

[3] IntServ. *Integrated services*. http://www.ietf.org/rfc/rfc1633.txt/. [Online; accessed 13 July 2012].

[4] DiffServ. *Differentiated services*. http://www.ietf.org/rfc/rfc2475.txt/. [Online; accessed 13 July 2012].

[5] M. Mihaylov, Y.-A. Le Borgne, K. Tuyls, and A. Nowé. *Decentralised reinforcement learning for energy-efficient scheduling in wireless sensor networks*. Proceedings of the International Journal of Communication Networks and Distributed Systems, 9:207–224, 2012.

[6] CTP. *Collection Tree Protocol*. http://www.tinyos.net/tinyos-2.x/doc/html/tep123.html/. [Online; accessed 13 July 2012].

[7] DYMO. *Dynamic MANET On-demand Routing Protocol*. http://tools.ietf.org/html/draft-ietf-manet-dymo-17/. [Online; accessed 13 July 2012].

[8] H. Zimmermann. *OSI Reference Model - The ISO Model of Architecture for Open Systems Interconnection*. IEEE Transactions on Communications, 28(4):425–432, April 1980.

[9] S. Kota, E. Hossain, R. Fantacci, and A. Karmouch. *Cross-layer protocol engineering for wireless mobile networks: Part 1*. IEEE Communications Magazine, 34(12):110–111, 2005.

[10] S. Kota, E. Hossain, R. Fantacci, and A. Karmouch. *Cross-layer protocol engineering for wireless mobile networks: Part 2*. IEEE Communications Magazine, 44(1):85–136, 2006.

3

Simulation and experimental validation of the QoS Framework

3.1 Introduction

The focus of this dissertation was on designing a QoS Framework including several QoS strategies that is able to improve QoS for wireless sensor networks that support Internet of Things applications. To this end, we have to implement the QoS Framework as presented in Chapter 2 and we have to evaluate and validate the implemented functionalities.

For this implementation, evaluation and validation, two separate paths are used during our research. On the one hand, we have implementation, evaluation and validation on real hardware. On the other hand, we have implementation, evaluation and validation using a simulator. For the former, the w-iLab.t testbed [2] with TmoteSky sensor nodes [1] is used, while for the latter, the OMNeT++ based Castalia simulator [3] is used. Both methods have advantages and drawbacks. Simulators have a big scalability and allow many different deployments in a short time, but real life channel and propagation models are often ignored and, as a consequence, the results are often not very realistic. On the other hand, real hardware results are very realistic, but it often takes a long time to test and debug different use cases and some scenarios, such as mobility, are harder to implement.

This chapter proceeds as follows. An overview of the network protocols that are used during our simulations and experiments is given in Section 3.2. In Section 3.3, the Castalia simulator is presented and the implementation of the QoS

Framework is discussed. The implementation of the QoS Framework on TmoteSky sensor nodes is on its turn discussed in Section 3.4. To this end, the IDRA architecture is presented and the integration with the QoS Framework is discussed. Furthermore, several experiments are performed and problems that we have encountered during these real-life experiments are discussed. We end this chapter with a conclusion in Section 3.5.

3.2 Network Protocols

Independent whether a layered, cross-layered or layerless sensor network architecture is used, some basic network protocols have to be chosen. In sensor networks, these are typically a physical protocol, a MAC protocol, a routing protocol and an application protocol. Depending on the requirements and the used architecture, one or more of these protocols can selected. For instance, the IDRA architecture allows using several MAC, routing and application protocols simultaneously.

In the following, we give an overview of the sensor network protocols that are used during this dissertation both in simulation and experimental validation.

3.2.1 Application Protocols

During this dissertation, three communication types will be mainly used: multipoint-to-point applications, multipoint-to-multipoint applications and point-to-multipoint applications. The specific operation of the application will be discussed during this dissertation.

3.2.2 Routing Protocols

For the multipoint-to-point application, the Collection Tree Protocol is used (CTP) [4], while for the point-to-multipoint and multipoint-to-multipoint applications, the Dynamic MANET On-Demand routing protocol (DYMO) [5] is used.

3.2.2.1 CTP Routing Protocol

Castalia implements a basic version of the Collection Three Protocol, named simpleTreeRouting. In this routing protocol, a sink node has to initiate a tree setup message that gets propagated through the network. Nodes that receive this packet take the sender as their parent node. When receiving a packet that has to be routed to the sink node, they forward it to their parent node. Several parameters can be adjusted to influence the parent selection, such as the maximum number of neighbors and the RSSI threshold before a node may become a neighbor.

3.2.2.2 DYMO Routing Protocol

For DYMO, an own implementation of the Dynamic MANET On-Demand (DYMO) routing protocol is used.

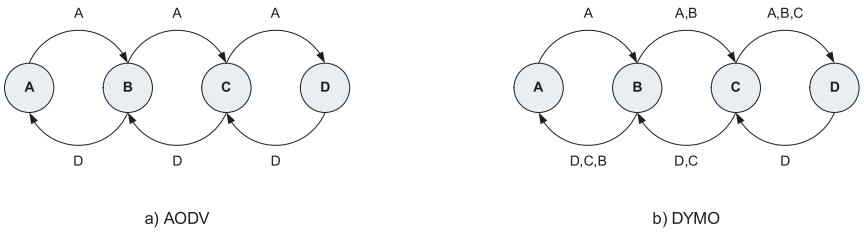


Figure 3.1. Comparison between AODV and DYMO

DYMO [5] is a point-to-point reactive routing protocol similar to the Ad hoc On-Demand Distance Vector (AODV) Routing protocol. Its operation is based on route request (RREQ), route reply (RREP) and route error (RERR) messages. The main difference between DYMO and AODV is that DYMO can append additional intermediate node information (see Fig. 3.1), and that its somewhat simpler design makes it more appropriate for an implementation on resource-limited devices such as sensor nodes. Our modification to DYMO lays in the fact that we only forward the RREQ message if our receiver quality is higher than a certain threshold (Eq. (3.1)). This way, we prevent that the network becomes flooded by unnecessary RREQ messages.

$$Prob_{forwarding} = Prob_{receiving\_RSSI>Threshold} \tag{3.1}$$

3.2.3 MAC Protocols

3.2.3.1 Always-On MAC

For the experimental validation, the CSMA-based always-on MAC that is available in IDRA is used. This means that the radio is always on and does not go to sleep mode, independent of transmit and receive behavior. Even though this implementation is not energy-efficient, it allows us to show the basic working of the QoS Framework.

3.2.3.2 LPL MAC

For the simulations, the low power listening (LPL) tunable MAC protocol that is available in the Castalia simulator is chosen as MAC protocol. The protocol operation for an LPL MAC with a duty cycle of 0.2 and listeninterval 100ms, is shown in Fig. 3.2. The duty cycle is defined as the time that the node is awake (listen period) to the total period of the node (listen period + sleep period).

Before node A transmits its data, it sends a Clear Channel Assessment (CCA) message to check if the channel is clear. If the medium is free, it starts periodically sending small beacon messages during a whole sleep interval, to be sure that each node (in the collision domain) is able to receive at least one beacon message. Afterwards, the actual data is sent. Node B is in the collision domain of node A, which means that node B is able to receive the beacon message. After that node B received this beacon message, it stays awake until the data transmission has finished. When the data transmission is finished, it waits a listen interval before going back to sleep because other packets can follow. Node C is not in the collision domain of node A, which means that it cannot hear the transmitted beacon messages. Because this node also has nothing to transmit, it proceeds with its normal sleep/wakeup scheme.

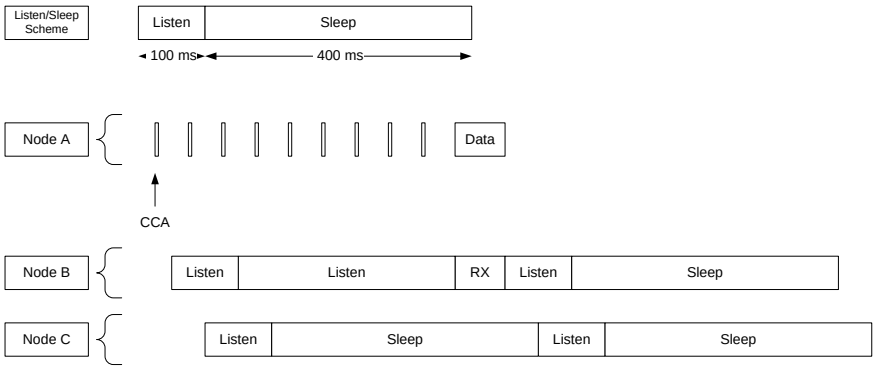


Figure 3.2. Operation of the tunable MAC protocol. Node A sends a data packet to node B by first performing CCA, afterwards sending beacon messages and finally sending the data packet. Node B stays awake when it receives one or more beacon messages until the data transfer is completed. Node C cannot receive these beacon messages and will again go to sleep after the listen interval.

3.2.4 Physical Protocol

For the physical layer, the CC2420 radio [6] is used. This radio has a theoretical bit rate of 250 kbps and an 8 MHz MSP430 microcontroller with 10k RAM and 48k Flash memory. The most important characteristics for this dissertation are summarized in Table 3.1.

Parameter	Value
Transmit Power	57.42 mW
Listen Power	62 mW
Sleep Power	1.4 mW
Receive Power	62 mW
Beacon size	16 bytes
CCA time	0.128 ms
Rx-Tx turnaround time	2*0.192 ms

Table 3.1. Radio settings

3.3 Simulation Architecture and Implementation

3.3.1 Castalia Network Simulator

Castalia is an OMNeT++ based simulator for wireless sensor networks, body area networks and generally networks of low-power embedded devices developed at NICTA, Australia.

OMNeT++ [7] is a public-source, component-based, modular and open architecture simulation environment with strong GUI support and an embeddable simulation kernel. Its primary application area is the simulation of communication networks and because of its generic and flexible architecture, it has been successfully used in other areas like the simulation of IT systems, queuing networks, hardware architectures and business processes as well. OMNeT++ is rapidly becoming a popular simulation platform in the scientific community as well as in industrial settings. Several open source simulation models have been published in the field of Internet simulations (IP, IPv6, MPLS, etc), mobility and ad-hoc simulations and other areas.

Castalia is very suited for simulating sensor networks since it implements several wireless channel and radio models. The most important advantages of this simulator are summarized below:

- Advanced channel model based on empirically measured data
- Advanced radio model based on real radios for low-power communication (e.g. CC2420 radio): multiple TX levels, switching between different power levels and simulating switching delay, flexible carrier sensing etc.
- Simulating node clock drift.
- Several MAC protocols available (e.g. a MAC protocol with a large number of parameters to tune).

- Several routing protocols available (e.g. a collection tree (CTP) routing protocol)
- Mobility support
- Designed for adaptation and expansion
- Good support

3.3.2 Implementation

3.3.2.1 Castalia Architectural Design

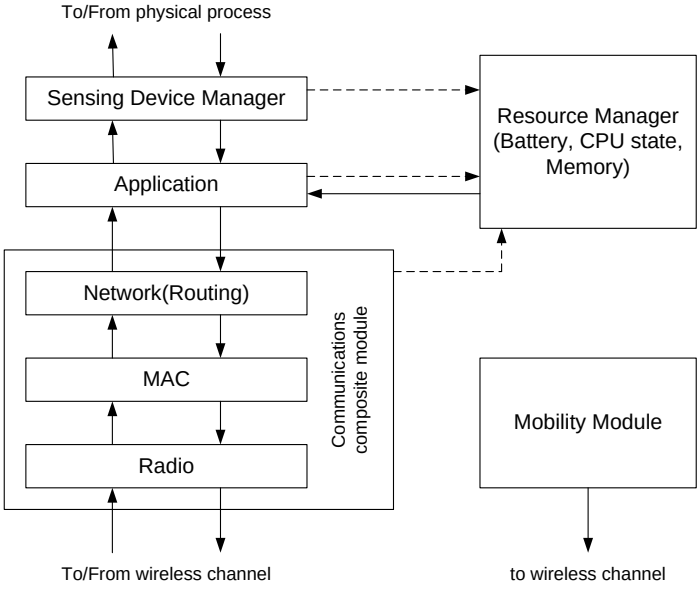


Figure 3.3. Castalia wireless sensor network simulator: node implementation

Fig. 3.3 shows the implementation of the node structure in Castalia. We can see from the figure that Castalia implements a layered network structure. The solid arrows represent messages that are passing, while the dashed arrows represent interfaces with simple function calls. The main architectural components are discussed below.

Communications Composite Module This module contains the network, MAC and Radio protocols. The radio module exchanges information with the wireless channel. Furthermore, these protocols exchange information with the resource manager (e.g. on the consumed energy).

Application This module implements the application. Castalia contains an application template which is easily adaptable by the application developer.

Sensor Device Manager The sensor device manager is responsible for simulating a truthful physical process, including the inaccuracies.

Resource Manager The resource manager keeps track of various node resources, of which energy is the most important. Almost every module exchanges information with this module on the consumed energy.

Mobility Module The mobility module specifies how the nodes move through space and it holds location states.

3.3.2.2 Protocol-Independent QoS Architectural Design

The modular design of Castalia easily allows cross-layer adaptation and QoS integration. This adaptation and integration is shown in Fig. 3.4.

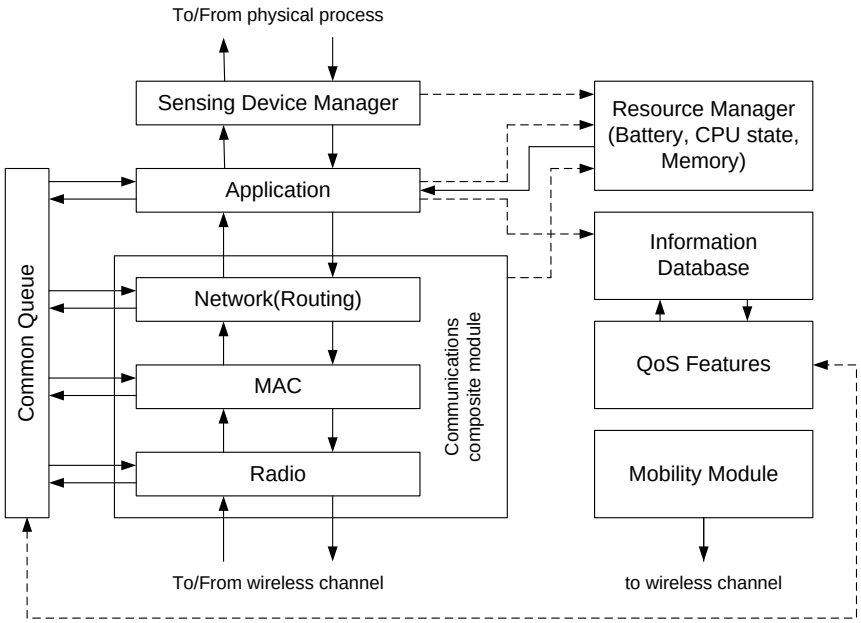


Figure 3.4. Castalia with QoS Framework integration: node implementation

In the following, the main interactions between the QoS Framework and the Castalia simulator are discussed. The sequence diagram of the protocol-independent QoS support is given in Fig. 3.5.

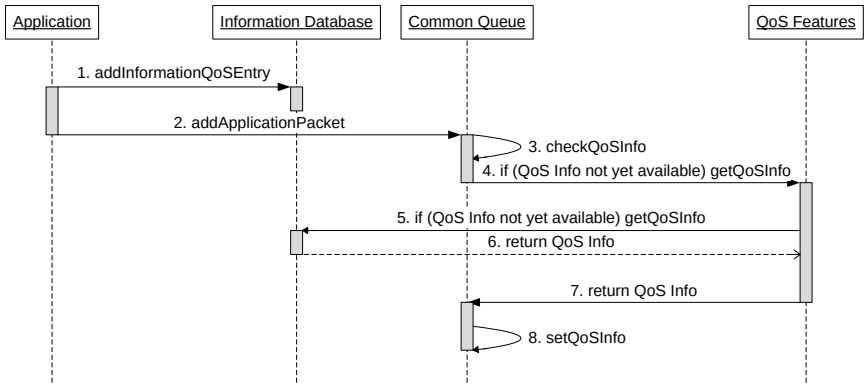


Figure 3.5. Sequence diagram of protocol-independent QoS support in Castalia

Common Queue Since Castalia implements a layered network architecture, each layer has its own storage queue. In our approach, we created one common queue which is used by the application, MAC, routing and radio layer to store their packets. This way, we always have an overview of the total number of packets in the system and of their QoS requirements.

Information Database The information database is used by the application developer to store metadata information on the exchanged data packets. For instance, information on the maximum end-to-end-delay, on the reliability, on the throughput and on the interpacket delay can be specified.

QoS Features The QoS Features contains the QoS Policies and QoS Monitoring and Management rules. They are currently integrated in one module for implementation simplicity. Furthermore, the QoS Features is responsible for the translation of application QoS requirements into network QoS requirements. The end-to-end delay has to be translated in single-hop delay etc.

3.3.2.3 Protocol-Dependent QoS Architectural Design

In Section 2.5.2, we have shown that the QoS Protocol Policies contain two functionalities: a Protocol Selector and a Protocol Parameter Configurator. Since Castalia doesn't allow dynamic protocol replacement, we have limited ourselves in the Castalia implementation to only support protocol parameter tuning.

To tune protocol parameters, several experiments have to be performed, with

different settings. However, simulations are very time-consuming, and performing many different simulations will take a long time to find the right parameters for a certain network scenario.

A solution to overcome these issues is using the SUMO toolbox [8]. This matlab toolbox allows to automatically build accurate surrogate models. By selecting a number of well chosen input parameters, an accurate output model is built. From this model, the representative output for each random input value can be observed, without simulating each input value separately.

Fig. 3.6 shows the coupling between the SUMO toolbox and the Castalia simulator. Since this SUMO Toolbox builds a metamodel of a given data source within the accuracy and time constraints set by the user, several parameters have to be configured.

First, in the SUMO Toolbox configure file, a metamodel type is selected. Examples are neural networks, kriging and polynomial functions. Furthermore, for input value selection, an initial design and a sample selector are chosen. Examples are a factorial design and a lola-voronoi sample selector. In a factorial design, an experiment consists of two or more factors, each with discrete possible values. In a full factorial design, the experimental units are all the possible combinations of these levels. In a fractional factorial design, some of these combinations are omitted. The lola-voronoi sample selector is a highly adaptive sampling algorithm which performs a trade-off between exploration (filling up the chosen design space) and exploitation (selecting data points in highly nonlinear regions).

When these parameters are configured, the SUMO Toolbox can do its work. For each input value that the SUMO Toolbox selects, the Castalia simulator is executed. This process continues until the target is reached. This target depends on the model parameter optimization, thus on how accurate the metamodel should be. This contains two parts, the quality estimation algorithm and the error function. An example of the former is 5-fold crossvalidation and an example of the latter is the root mean square error. More information on the SUMO Toolbox and how to use it can be found on the SUMO Toolbox website [9].

Fig. 3.7 illustrate the coupling between the SUMO Toolbox and Castalia. We have simulated the relation between the average time it takes for a packet to travel from source to destination (= data delay), the time that a route is valid (= route used time) and the packet rate (a packet rate of 2 here means that a packet is sent each 2 seconds). Fig. 3.7 gives the metamodel. Each black dot corresponds with the result of one DYMO simulation experiment. In our case we have a steep gap in the plot. Therefore we have chosen the sample selector in such a way that it chooses more points where there is more change in gradient and thus giving us more information about the abrupt region. Optimization is done using the optimization feature of the toolbox. From the figure, we see that when a route cannot be established and DYMO performs a route retry after 5 seconds (or 0.12 minutes), the average data

delay increases, which is in line with the expectations. It should be noted that this model was only an illustration of the coupling between the SUMO Toolbox and Castalia. Better metamodels could be built when the SUMO Toolbox configuration parameters are investigated and selected in more detail.

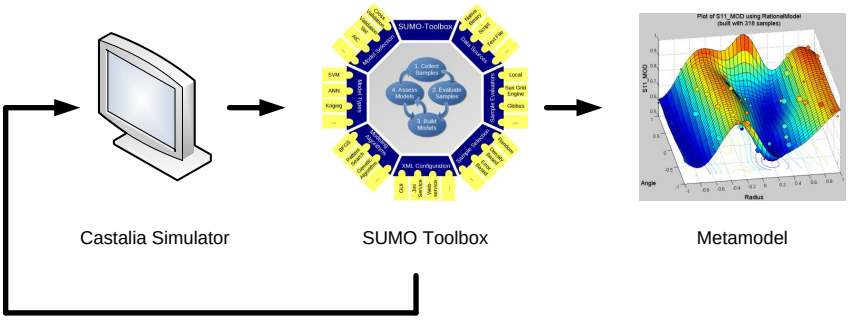


Figure 3.6. Coupling between the SUMO toolbox and the Castalia sensor network simulator

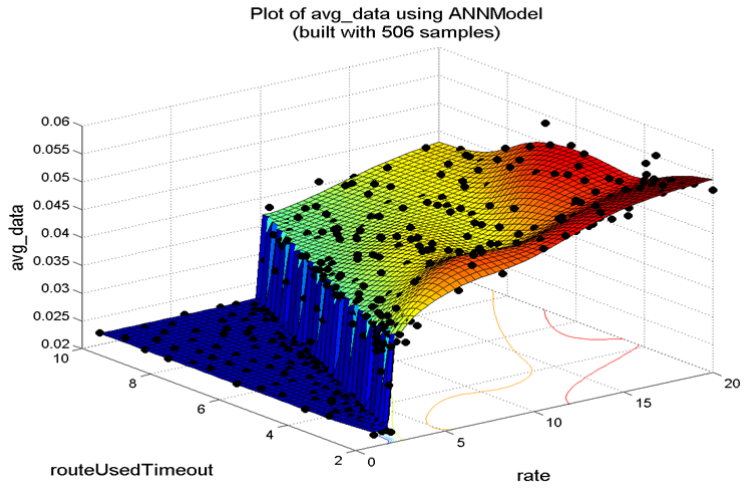


Figure 3.7. Metamodel plot of the average data delay with varying route used time out and packet rate.

3.4 Implementation and Experimental Validation on TmoteSky Sensor Nodes

3.4.1 w-iLab.t Testbed

The implementation of the QoS Framework on TmoteSky sensor nodes is done in TinyOS 2.1.0 and with nesC as a programming environment. The experimental validation is done on the sensor network part of the wireless w-iLab.t testbed. This testbed contains 200 TmoteSky sensor nodes, spread over three floors in a 100m x 15m office building. The w-iLab.t architecture is given in Fig. 3.8. Each sensor node is connected to an environment emulator (EE) and a small Alix computer (iNode), as can be seen in Fig. 3.8. The EE allows event emulation, e.g. sensor events or injection of audio. The iNodes are responsible for configuring the sensor node and are connected to a central management system [2].

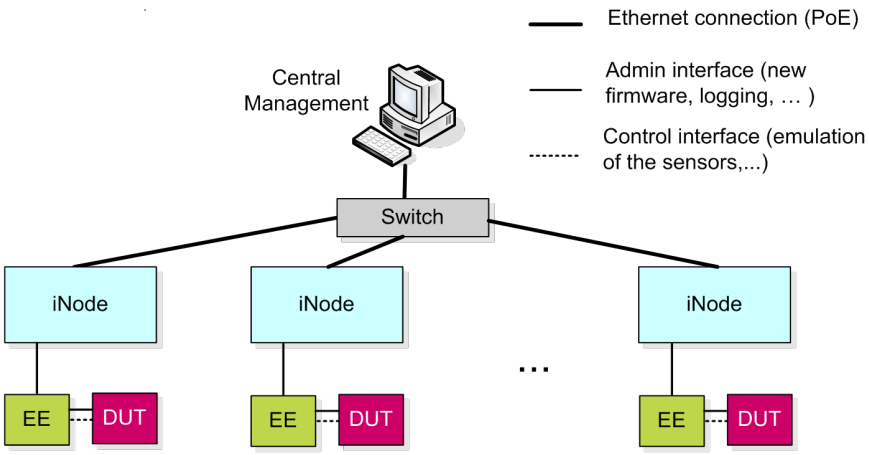


Figure 3.8. w-iLab.t architecture

The testbed allows simulating different topologies. To this end, the testbed contains nodes that are deployed in a square grid on the third floor, and nodes that are randomly deployed on the first and second floor. The topology of the deployed nodes is visualized in Fig. 3.9.

3.4.2 Implementation

3.4.2.1 IDRA Architectural Design

Recently E. De Poorter et al. have introduced a novel layerless sensor network architecture, called information driven architecture (IDRA) [10]. The IDRA architecture is not based on packets, but on *information exchanges*. Protocols do

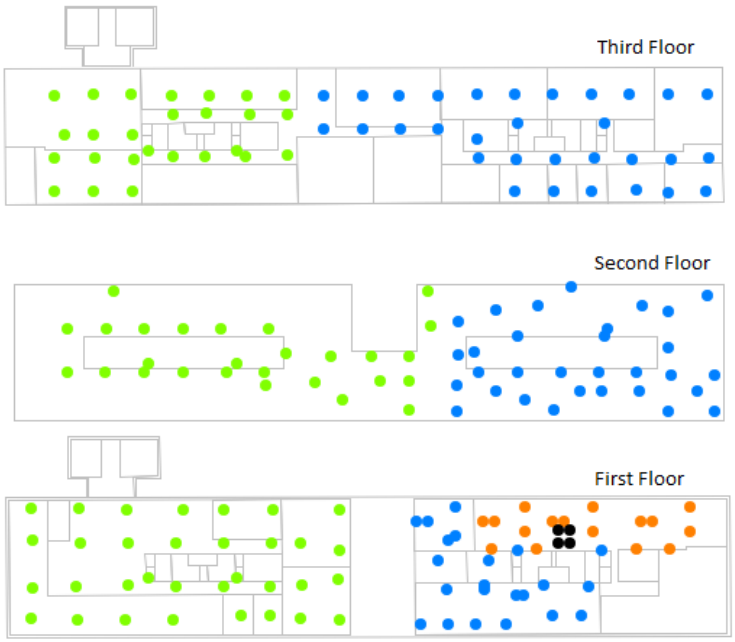


Figure 3.9. w-iLab.t topology

not interact with packets, but only with the information they contain, meaning that protocols do not need to define complicated header structures and do not need to provide buffer spaces for storing the packets. Instead, packet creation and buffer provisioning are handled by the architecture. A conceptual representation of this architecture is shown in Fig. 3.10.

The main characteristics of IDRA are:

1. The system architecture is responsible for packet creation. *Protocols only have to hand over their information to the system*. This avoids exchanging redundant information, allows information aggregation and the protocols do not have to care about header creation.

2. In contrast to traditional systems, where each protocol has to store-and-forward packets, *IDRA has only one system-wide queue* where incoming packets are stored. The advantages are less processing overhead, less memory usage, simpler monitoring and management, and protocols can be kept simpler and smaller.

3. *Protocol logic and packet representation are decoupled by a packet facade*. Since the system is responsible for packet creation, it is very easy to use a

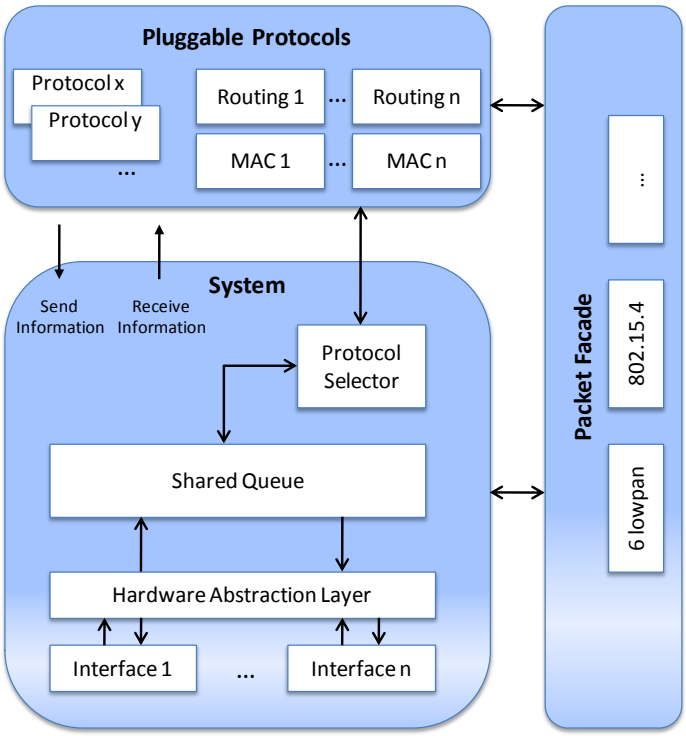


Figure 3.10. Information Driven Architecture (IDRA): conceptual representation

different packet implementation (802.15.4, 6lowpan, own implementation, etc.) which allows compatibility with legacy systems. Protocols only have to ask the desired information (for instance the destination) which is independent of the used packet implementation.

4. *The system decides at run-time which protocols have to be used.* This pluggable protocol system makes it possible to dynamically change between different routing and MAC protocols.

The main components of the IDRA architecture are discussed below:

Hardware Abstraction Layer Since sensor networks can be part of other wired and wireless, homogeneous and heterogeneous networks, QoS has to be delivered over the network's boundaries. Therefore, the *Hardware Abstraction Layer* is responsible for the communication with these networks.

Shared Packet Queue In contrast to traditional systems, where each protocol has to store-and-forward packets, IDRA has only one system-wide *Shared Queue*

where incoming packets are stored. As a consequence, information can be accessed, controlled and influenced at each network layer. This way, QoS decisions can be based on a global protocol stack view instead of a single layer protocol view.

Database & Packet Facade The *Database with Packet Facade* is responsible for storing and providing information. Since protocols only have to hand over their information to the system, IDRA itself is responsible for packet creation. This way, it is very easy to add protocol-independent QoS information to each packet. For instance, we can easily add a global priority level or some protocol-independent QoS attributes such as the information reliability or the maximum allowed information delay.

Pluggable Protocol System The system decides at runtime which protocols have to be used. This *Pluggable Protocol System* makes it possible to dynamically switch between different protocols such as routing and MAC protocols. This system allows the QoS framework to add or replace protocols on a per-need base. For instance, based on the node's capabilities and the network and application requirements, QoS functionality can be plugged in or out the system. Since there is no direct coupling between QoS and the network protocols, QoS can be simply enabled or disabled in the system depending on the application, the node and the network requirements. Basic QoS, such as packet priorities, can be enabled even if the protocols do not support any QoS features at all.

The Pluggable Protocol System can contain a ***QoS Monitoring*** module. Depending on the node's capabilities, it can monitor its own node's properties, its neighbors' properties or even the network's properties. Examples are the load, the remaining energy, etc.

IDRA is very suited to support QoS at an architectural level. The advantages are:

- *System-wide QoS*: since there is only one shared queue, information can be accessed, controlled and influenced at each network layer. This way, QoS decisions can be based on a global protocol stack view instead of a single layer protocol view.
- *Transparent QoS*: The packet facade and the information driven approach ensure that QoS information can be accessed in a transparent way. It makes it very easy to add protocol-independent QoS information such as a global priority level or protocol-independent QoS attributes such as the information reliability or the maximum allowed information delay.
- *Protocol-independent QoS*: Since there is no direct coupling between QoS and the network protocols, QoS can be simply enabled or disabled in the

system depending on the application and user requirements. Basic QoS, such as packet priorities, can be enabled even if the protocols do not support any QoS features.

- *QoS-aware data-aggregation*: Since protocols hand over their information to the system, it is much easier to take QoS requirements into account for information-aggregation. The system has a global view on which information has to be routed to which destinations under which QoS conditions and can easily interact with the QoS parameters such as delay and reliability. In a layered architecture, QoS-based information-aggregation can only be performed when packets are in their final sent queue.
- *Heterogeneous QoS support*: The pluggable protocol system allows the QoS system to add new, more optimized protocols on per-need base. Based on the node's capabilities, network and application requirements, QoS functionality can be plugged in or out the system.

3.4.2.2 Protocol-Independent QoS Architectural Design

Fig. 3.11 shows an overview of the implementation of the protocol-independent QoS support in the IDRA architecture. In the following, the main interactions between the QoS Framework and the IDRA architecture are discussed. The sequence diagram is given in Fig. 3.12.

Application/User QoS Requirements The application developer is responsible for registering the application requirements in the Information Database (see Fig. 3.12, 1. addInformationParameter and 2. setQoSParameter) in terms of end-to-end delay, throughput, packet loss rate and reliability and the QoS Framework will translate these requirements into network requirements expressed in terms of single-hop delay, node throughput, packet loss rate and packet delay variation. From then, the application is ready to send packets with the requested QoS level (see Fig. 3.12, 5. sendParameter)

Network Application Aggregator Sensor networks will support diverse applications with different QoS requirements on delay, reliability, jitter and bandwidth. Examples are reliable eHealth monitoring applications (e.g. blood pressure), time-critical and bandwidth consuming streaming applications (e.g. emergency voice call), best effort monitoring application (e.g. environmental monitoring) etc. The *Network Application Aggregator* is responsible for managing and controlling these applications. For instance, it aggregates the individual QoS requirements of the applications in general node and network requirements and it checks if the applications are allowed on the network. Furthermore, it can give feedback to the applications on the requested QoS level.

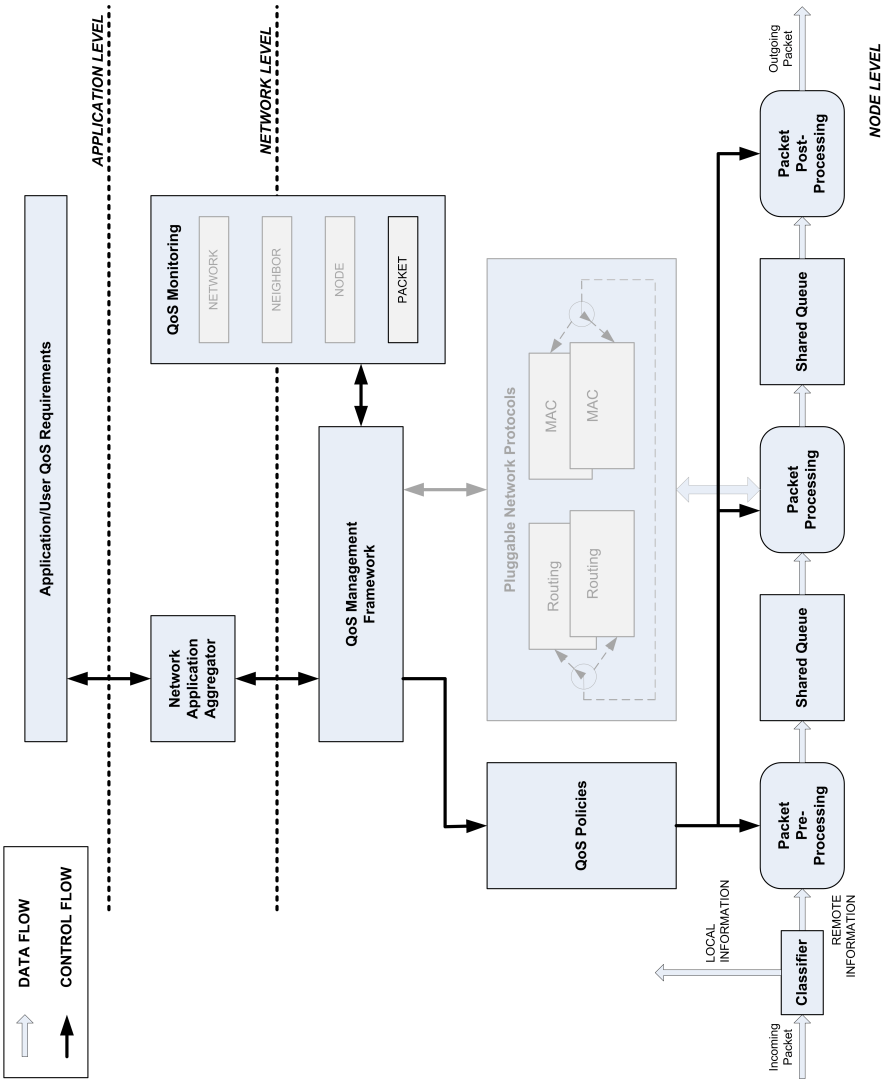


Figure 3.11. Protocol-independent QoS support in IDRA

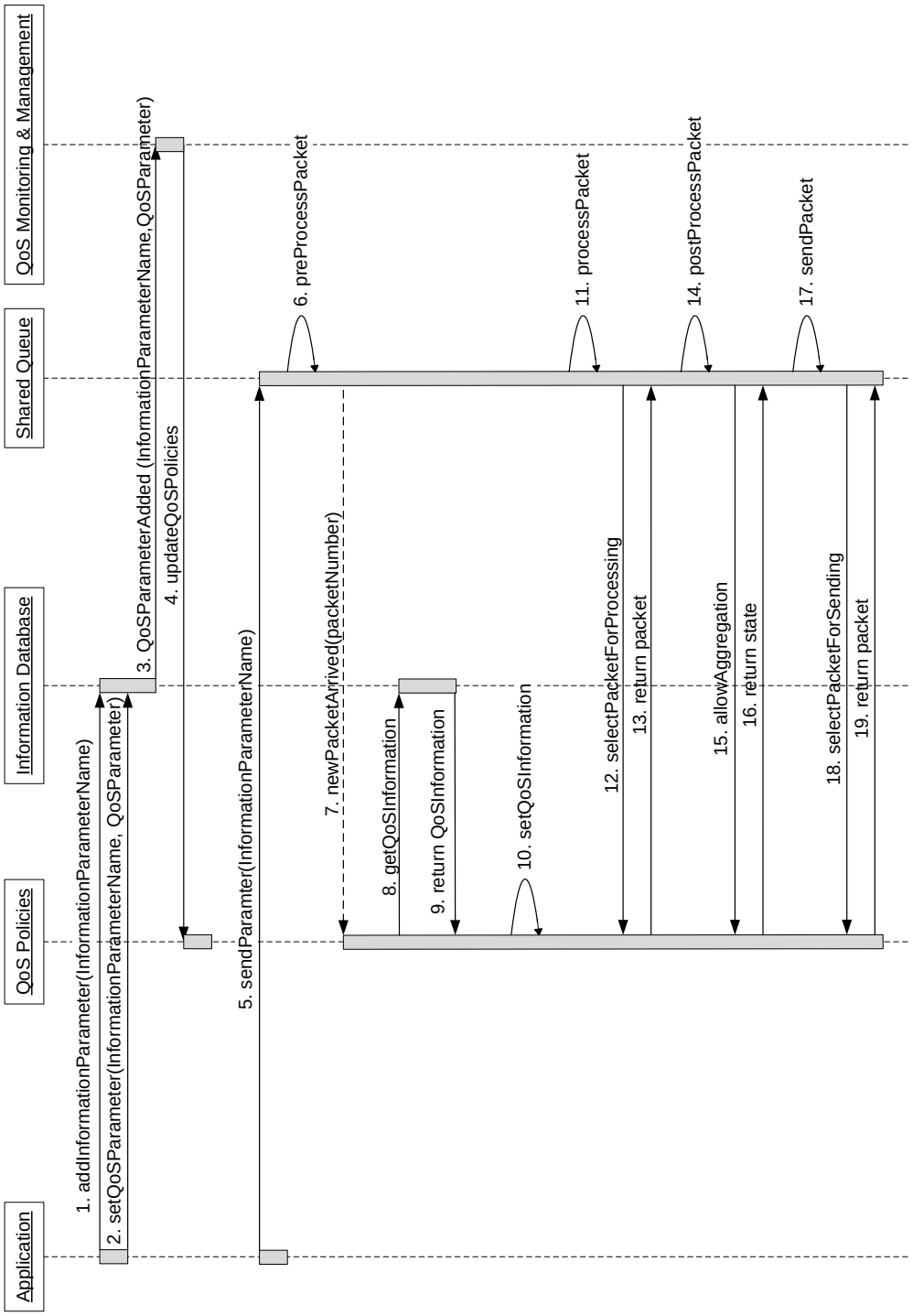


Figure 3.12. Sequence diagram of the protocol-independent QoS support in IDRA

QoS Monitoring and Management When an application has added or updated QoS information, the QoS Monitoring and Management module is informed. Based on this information, the QoS Policies can be updated. For example, adding a new voice application with stringent delay constraints can change the priority level of an existing voice application which still has enough time left.

QoS Packet Policies The QoS Policies, shown in Fig. 3.13, define the internal rules for processing information through the system. These rules are working on three levels:

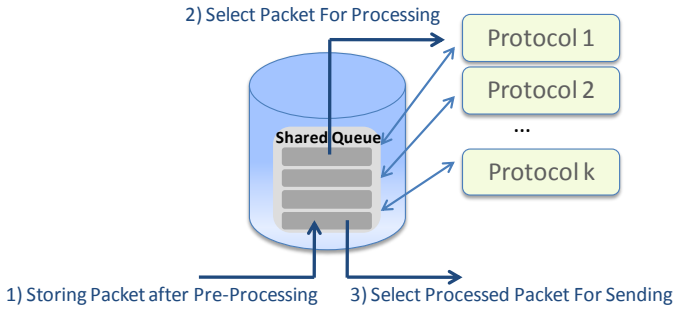


Figure 3.13. Conceptual representation of the QoS Policies

1. When a packet arrives in the Shared Queue, IDRA executes packet pre-processing. This will inform the QoS Policies that a new packet has arrived. When this packet is a new packet, the QoS Policies is responsible for setting the initial priority level and QoS attributes, based on the information that it can find in the Information Database. Furthermore, if the shared queue is almost full, a packet can be dropped in order to keep free spaces for newly arriving packets. The QoS Policies will define which packet has to be dropped first: the reliable monitoring packet with the less stringent delay requirements or the voice packet with the lowest reliability level but with the most stringent delay requirements, or should it be a combination of both? Furthermore, at this stage, some minor modifications can be made to each packet. For instance, if our voice packet is processed, we can update the current delay that our packet has already undergone. Even the priority level can be changed at this stage. Suppose there are two voice calls with the same priority level. The maximum delay of the packets of the first voice call is almost reached, while the packets of the second voice call only have experienced a minor delay. In this case, the QoS Policies can decide to increase the priority level of the first voice call.

2. When the system is ready to process a new packet, the QoS Policies select the packet that has to be processed first. Some packets, such as control and management messages always need to be processed first in order to keep the network alive. But for data packets, this packet selection process can be smartly controlled.

3. After a packet has been processed, control is again handed over to the Shared Queue. The post-processing will check if and how aggregation should be performed. Finally, when the MAC is ready, the QoS Policies can select the packet that should be sent first.

3.4.2.3 Protocol-Dependent QoS Architectural Design

Fig. 3.14 shows an overview of the implementation of the protocol-dependent QoS support in the IDRA architecture. In the following, the main interactions between the QoS Framework and the IDRA architecture are discussed. The sequence diagram is given in Fig. 3.15.

QoS Monitoring and Management The QoS Monitoring and Management module is responsible for monitoring the information that is needed for the protocol selection and protocol parameter tuning. It will be informed when network information (e.g. density, path reliability), neighbor information (e.g. remaining energy level, mobility), node information (e.g. energy level) or application information changes. Based on this information, the appropriate network protocols are selected or adapted.

Pluggable Network Protocols The Pluggable Network Protocols contains the available network protocols. For each *group* of protocols, a pluggable QoS module is available. For the MAC protocols, this is the MACQoS module, while for the Routing protocols, this is the RoutingQoS module. The QoS Monitoring and Management module interacts with these modules. When an application is added or changes one of its parameters, this can have influences on the current setting of MAC and Routing. For instance, the MAC module can adjust its time slots when a higher throughput is required or the Routing module can change its routing path due to mobile nodes. The management module will act on these changes by fitting the network protocols to the right settings.

3.4.3 Experimental Evaluation and Validation

In the following, we evaluate the implementation of the QoS Framework on w-iLab.t.

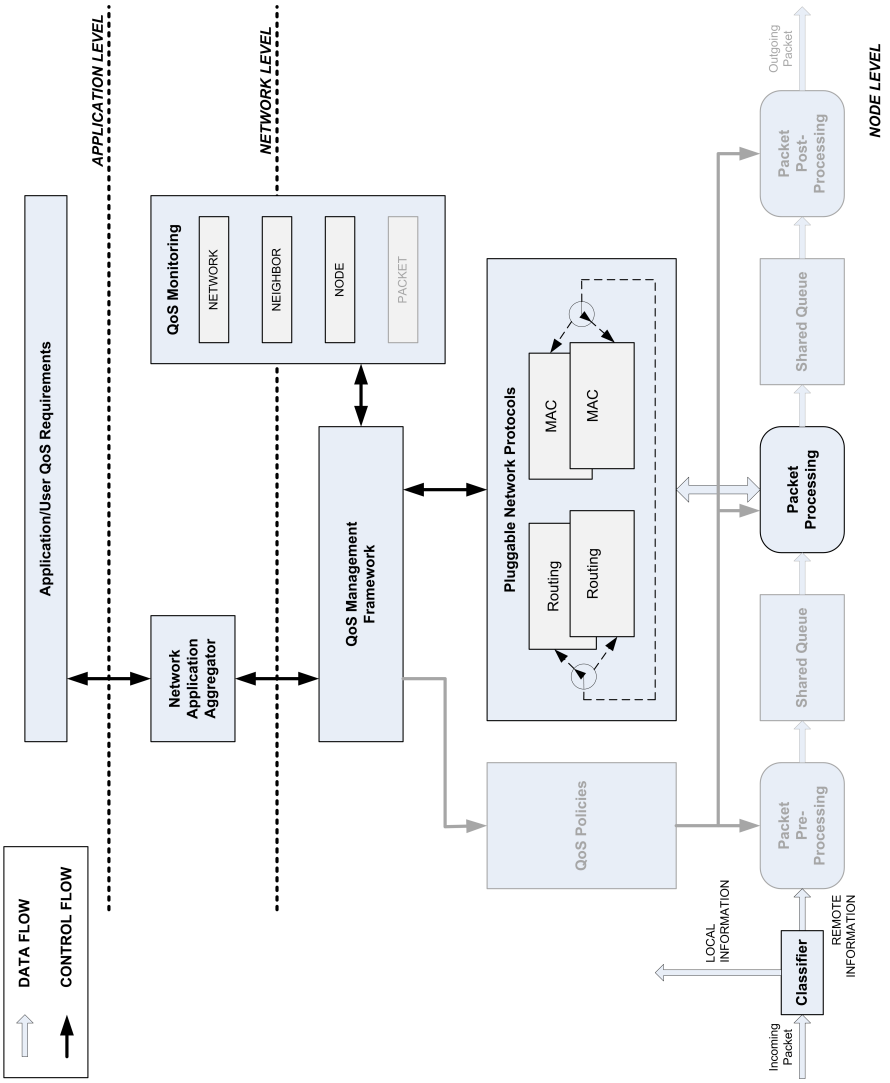


Figure 3.14. Protocol-dependent QoS support in IDRA

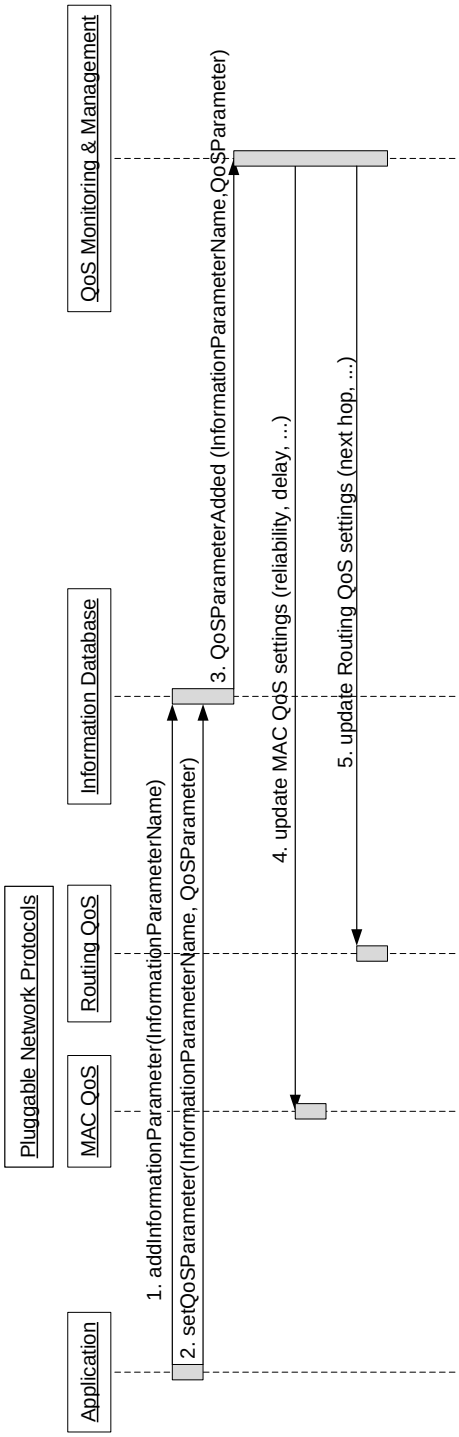


Figure 3.15. Sequence diagram of the protocol-dependent QoS support in IDRA

The implementation of the QoS Framework is tested for both the DYMO routing protocol and the CTP routing protocol.

In our experiment, half of the third floor of the w-iLab.t testbed is used. As a MAC protocol, the always-on MAC protocol is used, which checks periodically if it has packets to send.

3.4.3.1 Evaluation and Validation with DYMO Routing Protocol

For the DYMO protocol evaluation, we used half of the third floor of the w-iLab.t testbed. In the following, the testbed results are discussed in two scenarios. In the first scenario, two traffic flows with QoS support are considered: 1 high priority traffic flow and 1 low priority traffic flow. In the second scenario, two traffic flows without QoS support and thus with the same priority level are considered. For both scenarios, some throughput/drop results will be discussed.

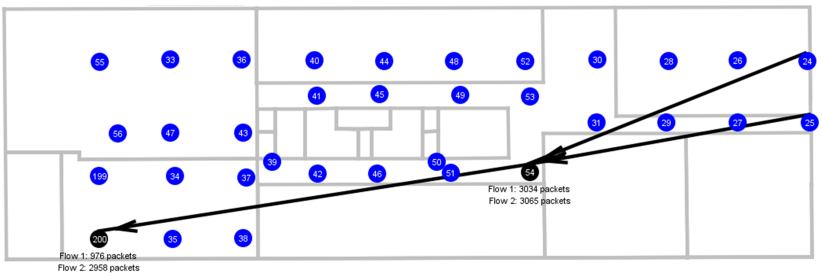


Figure 3.16. With QoS support: adding a high priority data flow

Scenario 1: with QoS Support In this first scenario, two traffic flows with QoS support are considered. To illustrate this QoS support, two traffic flows with different priority levels are used. Both traffic streams will send a packet of 70 bytes payload every 150ms and each node checks every 100ms if it has a packet to send. At the beginning of the experiment, there is only 1 low priority traffic flow (flow 1) between sensor nodes 25 and 200. After a while, a high priority traffic flow (flow 2) is set up between node 24 and 200. As can be seen in Fig. 3.16 both flows meet each other at node 54. Since more packets arrive at node 54 than it can process, some packets will have to be dropped. The collected database results show that at the end of the experiment 2017 packets from the low priority traffic flow were dropped while 0 packets from the high priority traffic flow were dropped. These results are also shown in the left part of Fig. 3.18.

Scenario 2: without QoS Support In this second scenario, two traffic flows without QoS support are considered. In the QoS Framework, this scenario can be

simulated by using two traffic flows with the same priority level. As in scenario 1, one traffic flow starts sending between node 25 and 200. After a while, the second traffic flow with the same priority level is set up between node 24 and node 200. This time, the collected database results show that 1098 packets from the first traffic flow were dropped while 1100 packets from the second traffic flow were dropped (Fig. 3.17). The right part of Fig. 3.18 shows these results.

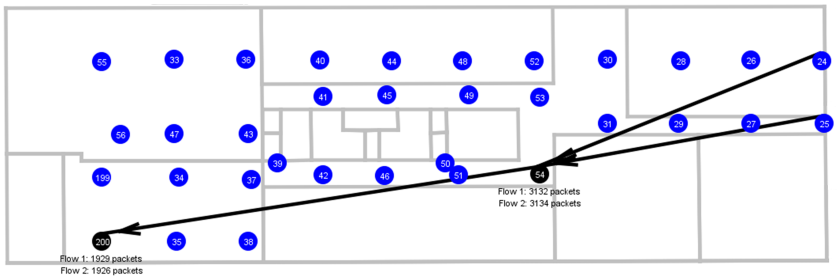


Figure 3.17. Without QoS support

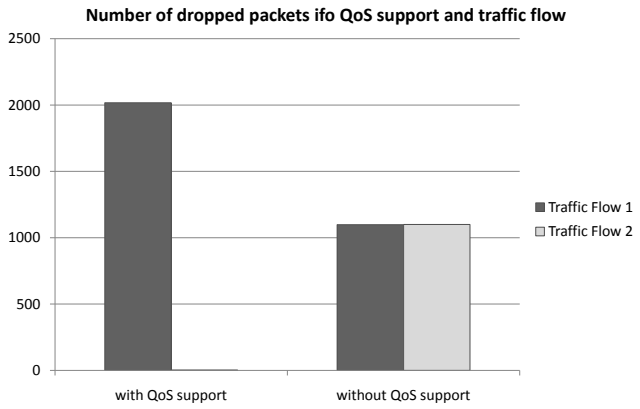


Figure 3.18. Number of dropped packets with and without QoS support

3.4.3.2 Evaluation and Validation with CTP Routing Protocol

For the CTP protocol evaluation, we used half of the second floor of the w-iLab.t testbed. Again, the testbed results are discussed for two scenarios. In the first scenario, one node generates a high priority traffic flow towards the sink node, while

the other nodes only send low priority traffic. In the second scenario, all nodes are sending traffic with the same priority level. For both scenarios, some throughput/-drop results will be discussed. Furthermore, since this scenario contains more than one intermediate hop, the delay results will also be discussed.

Scenario 1: with QoS Support As can be seen in Fig. 3.19, node 89 was configured as an always on sink node. The other nodes were generating data packets every 4 seconds but, due to their sleeping scheme, they could only transmit a packet every second. This way, we created an overloaded network where packets had to be dropped. Two types of traffic were generated, each having a different priority level. The high priority traffic flow was generated by a single node (Node 107), while all the other nodes were generating a low priority traffic flow.

Statistics have been analyzed for all nodes that are on the same routing level as node 107, i.e. 5 neighbors away from the sink node.

Experimental results show that node 89 receives 828 out of 843 sent packets from node 107, while node 89 only receives 19, 7 and 69 packets from nodes 106, 108 and 109 respectively. While the average reliability of the 1-hop neighbors of the sink node was still 98.47%, this is reduced to 28% for the 4-hop neighbors. The QoS support is able to increase the reliability for the 5-hop neighbor 107 up to 98.22%, while the other 5-hop neighbors only have a reliability of 3.75%.

Delay results show that the 1-hop neighbors of the sink node encounter an average end-to-end delay of 3.1 seconds, which increases up to 13.12 seconds for the 4-hop neighbors. The QoS support prioritized the high-priority stream of node 107, which results in an end-to-end delay of 1.58 seconds compared to an average value of 17.59 seconds for the other 5-hop neighbors.

These results are summarized in Table 3.2.

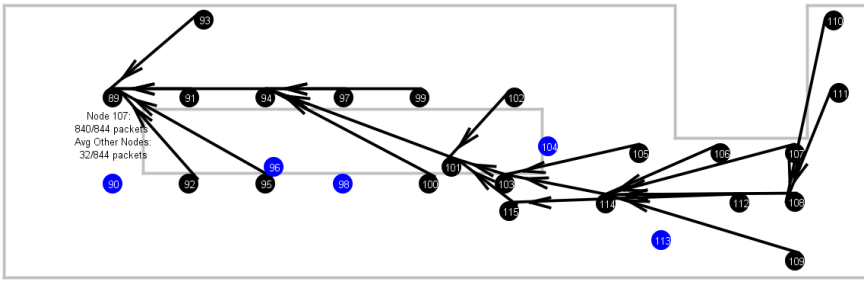


Figure 3.19. With QoS support

Scenario 2: without QoS Support In this second scenario, all nodes generate traffic with the same priority level towards the sink node (node 89). To compare the results 'without QoS support' with the results 'with QoS support', statistics are analyzed for all nodes that are on the same routing level as node 107 (which now generates traffic with the same priority level as the other nodes), i.e. 5 neighbors away from the sink node.

Experimental results show that node 89 receives 132, 109, 44 and 72 out of an average of 843 sent packets per node for nodes 107, 106, 108 and 109 respectively. While the average reliability of the 1-hop neighbors of the sink node still was 89.56%, this is reduced to an average of 8.90% for the 5-hop neighbors. The reliability of node 107 was 15.66, which is a little bit higher than the average value.

Delay results show that the 1-hop neighbors of the sink node encounter an average end-to-end delay of 1.07 seconds, which increases up to 14.37 seconds for the 5-hop neighbors. The end-to-end delay of node 107 was 14.33 seconds, which is a little bit slower than the average value.

These results are summarized in Table 3.2. We can conclude that the delay and packet loss for the high priority stream is significantly lower, even though the network protocols do not support any QoS at all.

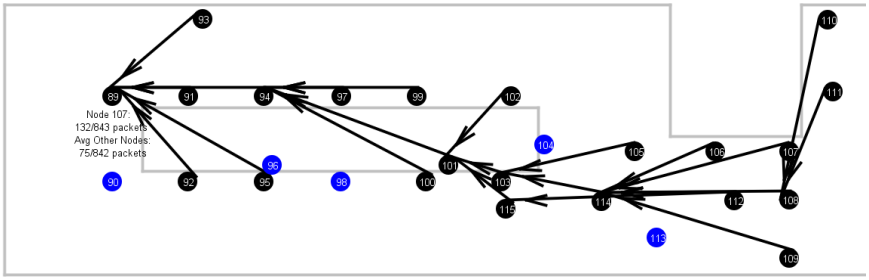


Figure 3.20. Without QoS support

Memory Footprint Table 3.3 shows the memory footprint of the QoS modules compared to the memory footprint of the other IDRA system modules. One of the characteristics of IDRA is its low protocol memory cost, at the price of a somewhat bigger initial memory cost. When comparing the QoS system (with QoS Policies, QoS Management and QoS Application Database) in IDRA with the total IDRA system using a simple MAC protocol and a broadcast routing protocol (Basic IDRA system + Broadcast Routing + Simple MAC protocol + QoS system),

	Without QoS Support	With QoS Support
	Average end-to-end delay	
Low-priority traffic flow	14.38 s	17.59 s
High-priority traffic flow	14.33 s	1.58 s
	Average reliability	
Low-priority traffic flow	8.90%	3.76%
High-priority traffic flow	15.66%	98.22%

Table 3.2. CTP QoS analysis

Module	ROM (bytes)	RAM (bytes)
IDRA system	20236	5344
Broadcast Routing	390	111
DYMO Routing	5008	312 (+18 per route)
CTP Routing	712	130
Simple MAC protocol	844	24
Advanced MAC	7136	1264
Neighbor database	8536	2631
QoS Policies	1816	10
QoS Management	3072	238
QoS Application Database	4068	668

Table 3.3. Memory Footprint

the total QoS architecture takes about 29% of the overall ROM memory usage and about 14% of the overall RAM memory usage. This is not negligible and it is the price we pay for a better overall QoS. However, if we compare our QoS architecture with an IDRA system with more advanced modules, for instance when taking into account the basic IDRA system with the DYMO routing in combination with the Advance MAC protocol with its own Neighbor database, the absolute QoS footprint remains unchanged, while the relative QoS footprint is decreased to 18% of the overall ROM memory usage and 9% of the overall RAM memory usage. We note that the implementation of the QoS architecture is protocol-independent.

3.4.4 Problems Encountered

When performing real-life experiments, we encountered several problems since we had to push the limits of the node capabilities as every node was also burdened with the task of collecting results via the serial interface, on top of regular packet forwarding.

Firstly, we encountered a problem with the MSP430 CPU @ 8 MHz. When two interrupts were triggered within a time frame of 8 system clock cycles (which

fits exactly our definition of pushing the limits on the mote), the CPU might fetch a random address if the system clock is greater than 6 MHz. To overcome this issue, we had to reduce the system frequency below 6 MHz.

Secondly, the serial stack driver was malfunctioning due to the use of atomic blocks. When entering an atomic block, drivers were put into a coma because all interrupts were disabled. Furthermore, when a serial frame (transmitted at 115200 baud) was received by the sensor node, the serial bytes were arriving at a rate of one byte every 87 μs (8 bits + 1 start-bit + 1 stop-bit @ 115200 bits/s) which resulted in an interrupt every 87 μs. So if the system was blocked around 90 μs by other interrupts and atomic blocks then you may lose a byte. If you lose a byte, the frame cyclic redundancy check (CRC) fails and the frame is dropped. This explains why the serial driver is sensitive to atomic blocks, but the other drivers are also affected. To resolve this issue, the serial stack needs to be reimplemented.

3.5 Conclusion

In this chapter, we have shown how our developed QoS Framework can successfully be implemented in the Castalia simulator and on TmoteSky sensor nodes.

Firstly, we have extended the layered Castalia simulator to allow several cross-layer interactions. This implementation will be used for QoS-aware in network aggregation in Chapter 5 to Chapter 7. Furthermore, we have shown how both the protocol-independent QoS support and the protocol-dependent QoS support can be integrated. For the latter, a coupling between the Castalia simulator and the SUMO toolbox was made and this was illustrated with a scenario in which DYMO has to perform a route retry.

Secondly, we integrated the QoS Framework in the IDRA sensor network architecture on TmoteSky sensor nodes. The implementation was evaluated and validated through several experiments. We have shown that by prioritizing packet streams the delay of a high priority stream can be reduced significantly (in our CTP experiment from 14.33 s to 1.58 s) and that the reliability can be improved (in our CTP experiment from 15.66% to 98.22%). This comes at the cost of a slightly higher delay for the non prioritized traffic stream (in our CTP experiment from 14.38 s to 17.59 s) and a little reduced reliability (in our CTP experiment from 8.90% to 3.76%). Furthermore, we have shown that the total QoS architecture takes about 29% of the total IDRA system's ROM memory and about 14% of the system's RAM memory. This is not negligible and it is the price we pay for a better overall QoS. However, it should be noted that when the QoS Framework is implemented in a more complex IDRA system, the relative QoS footprint is decreased to 18% of the system's ROM memory and 9% of the system's RAM memory.

References

[1] Tmote Sky Datasheet. *Ultra low power IEEE 802.15.3 compliant wireless sensor module*.

[2] L. Tytgat, B. Jooris, P. De Mil, B. Latré, I. Moerman, and P. Demeester. *Demo abstract: WiLab, a real-life Wireless Sensor Testbed with Environment Emulation*. In European conference on Wireless Sensor Networks, EWSN adjunct poster proceedings, February 2009.

[3] Castalia. *A simulator for wireless sensor networks*. http://castalia.npc.nicta.com.au/. [Online; accessed 13 July 2012].

[4] CTP. *Collection Tree Protocol*. http://www.tinyos.net/tinyos-2.x/doc/html/tep123.html/. [Online; accessed 13 July 2012].

[5] DYMO. *Dynamic MANET On-demand Routing Protocol*. http://tools.ietf.org/html/draft-ietf-manet-dymo-17/. [Online; accessed 13 July 2012].

[6] Texas Instruments. *CC2420 Radio*. http://www.ti.com/product/cc2420/. [Online; accessed 13 July 2012].

[7] OMNeT++. *Network Simulation Framework*. http://www.omnetpp.org/. [Online; accessed 13 July 2012].

[8] D. Gorissen, K. Crombecq, I. Couckuyt, T. Dhaene, and P. Demeester. *A Surrogate Modeling and Adaptive Sampling Toolbox for Computer Based Design*. Journal of Machine Learning Research, 11:2051–2055, July 2010.

[9] SUMO. *Surrogate Modeling Toolbox*. http://www.sumowiki.intec.ugent.be/. [Online; accessed 13 July 2012].

[10] E. De Poorter, I. Moerman, and P. Demeester. *An Information Driven Sensornet Architecture*. In Proceedings of the 3nd International Conference on Senor Technologies and Applications, June 2009.

4

In-network Aggregation in Wireless Sensor Networks

4.1 Introduction

A typical wireless sensor network is equipped with low-cost and low-power sensor nodes which are often battery powered. Frequent replacement of batteries should be avoided as much as possible, certainly when they are deployed in hard to reach locations. A technique that is often used to reduce energy consumption in wireless sensor networks is in-network aggregation.

In-network aggregation is in fact a very broad concept. It can be defined as follows [1]:

> "In-network aggregation is the global process of gathering and routing information through a multi-hop network, processing data at intermediate nodes with the objective of reducing resource consumption (in particular energy), thereby increasing network lifetime."

There are many in-network aggregation approaches. An overview is given in Fig. 4.1.

- *In-Network Packet Aggregation.* In this approach, multiple packets are aggregated into one. Whole packets (including their header) are aggregated into a new packet with a new header.

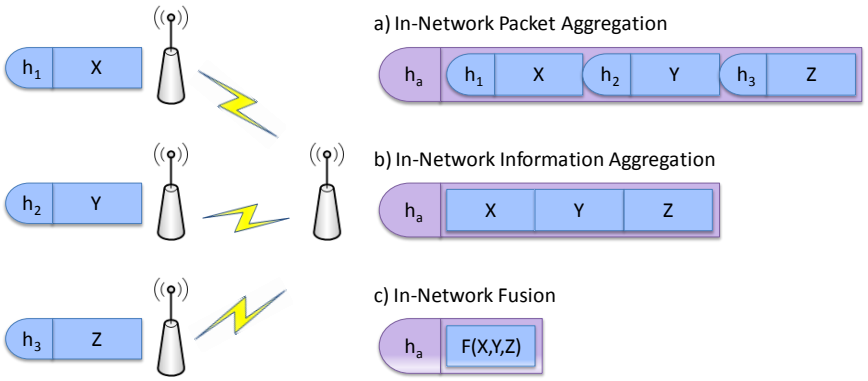


Figure 4.1. Classification of In-Network Aggregation Approaches

- *In-Network Information Aggregation*. When performing information aggregation, the packets' header is removed and only the information part is integrated in a new packet with a new header.

- *In-Network Fusion*. With fusion, data collected from one or more sources is typically processed by an arithmetic function such as MIN, MAX or AVG before sending the collected information through the network. Fusion can be seen as a mapping of several objects to a single object in an optimal fashion [2].

The latter approach is in literature also known as lossy aggregation because the original packets cannot be reconstructed, while the former two approaches are also known as lossless aggregation.

In this thesis, in-network packet aggregation is considered. The in-network aggregation is performed in each node in the same manner. The number of packets that are aggregated in a single packet is in the following referred to as the degree of aggregation (DoA).

The remainder of this chapter is structured as follows. In Section 4.2, an overview of the ongoing research on in-network aggregation is given and in Section 4.3 the impact of in-network aggregation on energy consumption and QoS metrics is investigated. Next, in Section 4.4, we show how in-network aggregation can be performed in the Internet of Things with standardized network protocols and in Section 4.5, we show how in-network aggregation can be performed in our QoS framework. Finally, we end this chapter with a Conclusion in Section 4.6.

4.2 Related Work

Originally, in-network aggregation was proposed as a technique to combine packets (coming from different nodes) that are routed towards the same destination(s). As a consequence, it is straightforward to classify in-network aggregation protocols similarly to wireless sensor network routing protocols according to their underlying network structure: flat, hierarchical or location-based.

A first category is *flat network* data-centric in-network aggregation.In *flat* in-network aggregation, all nodes play identical roles. This type of aggregation is often accomplished by data-centric routing in which a sink node transmits a query message and sensor nodes that have the requested data respond back to the sink. Typical examples are SPIN [3] and Directed Diffusion [4].

In *hierarchical* in-network aggregation, aggregation is performed at special nodes. We can make a distinction between three approaches. In a tree-based approach, aggregation is performed at intermediate nodes along the aggregation tree (e.g. EADAT [5] in which an aggregation tree is constructed based on the residual power). In a chain-based approach, each node only transmits to its closest neighbors (e.g. PEGASIS [6] in which in each data-gathering round, a node receives the data from one of its neighbors and transmits the aggregated data to its other neighbor along the chain). Finally, in a cluster-based approach, nodes send their data to a cluster head that in turn transmits the data to the sink (e.g. LEACH [7]). Hybrid approaches are also possible, such as in Tributaries and Deltas [8], which combines an aggregation tree under low packet loss rates and a multipath approach in case of high packet loss rates.

In *location-based* in-network aggregation, location information is used to perform in-network aggregation. An example can be found in [9], in which information within a certain grid is sent to the data aggregator of that grid.

Some in-network aggregation approaches cannot be classified according to their underlying network structure but depend on their protocol operation. For instance, with *network-flow-based-protocols*, the sensor network is represented as a graph and in-network aggregation is modeled as a network flow problem [10].

4.3 Impact of In-Network Aggregation

In this section, the impact of in-network aggregation on the energy consumption and on the QoS metrics, delay and reliability, is investigated. As stated before, sensor nodes have scarce resources and saving energy is an important issue in sensor networks.

4.3.1 Impact on Energy Consumption

Fig. 4.2 shows the current consumption of the Zolertia Z1 node [11]. This node has a CC2420 radio with a theoretical bit rate of 250 kbps and a 16 MHz MSP430 microcontroller with only 8k RAM and 92k Flash memory. It can be seen that most of the energy consumed by sensor nodes is due to the radio transmission and reception. As a consequence, minimizing the number of radio transmissions and receptions can increase the radio sleep time, which reduces the total power consumption significantly and thus increases the overall network's lifetime.

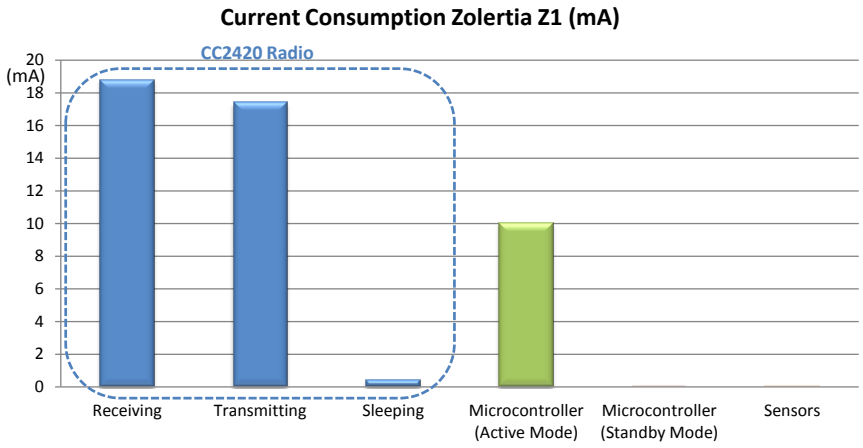


Figure 4.2. Current consumption Zolertia Z1

The following section explains in more detail why reducing the number of radio transmissions can lead to reduced power consumption. For the analysis, the CC2420 radio [12] and the IEEE 802.15.4 standard [13], which defines the physical and media access control (MAC) layers for low-rate wireless personal area networks (LR-WPAN) such as wireless sensor networks, are used. Within this standard, a physical data rate of 250 kbit/s and a maximum packet size of 128 bytes are defined.

The total transmission energy consumption (E_{tx}) by the CC2420 radio communication can be described as follows:

$$E_{tx} = N_{tx}(P_{tx}T_{tx} + P_{st-tx}T_{st-tx}) \tag{4.1}$$

with:

N_{tx} : number of transmissions
P_{tx} : the power consumed by the transmitter during transmission
T_{tx} : the transmit time
P_{st-tx} : the power consumed by the transmitter during startup
T_{st-tx} : the startup time needed for a transmission

From Eq. 4.1, it is clear that there are three important factors that have an impact on the energy consumption:

- The number of transmissions
- The transmit energy
- The startup energy

Firstly, the startup energy is fixed for each transmission, so reducing the number of transmissions will reduce the total startup energy. The startup time is the time that the transceiver spends to wake up from sleep mode to transmit mode and switch back to receive mode (RX-TX turnaround time, 2x 192 μs) prior to transmitting or sleeping again. When the startup time approximates the transmit time, which is the case for small packets, this can have a big impact on the total energy consumption. The time to bring the crystal oscillator up is ignored because this is performed at a lower power level.

Additionally, it is very common that a radio performs a clear channel assessment (CCA) prior to sending. When the channel is not free, the radio performs a backoff for some short, random period before attempting to transmit again. Although CCA can be disabled, it is performed by default by the CC2420 radio. This lasts 8 symbol periods or 128 μs.

Furthermore, prior to sending the transmit data, a Synchronization Header (SHR) is sent. This SHR contains a physical dependent preamble sequence and Start-of-Frame Delimiter (SFD). According to the 802.15.4 standard, this SHR lasts 10 symbol times or 160 μs.

Combining the above the RX-TX turnaround time, CCA time and SHR, the total 'startup overhead' lasts 672 μs. In this time, approximately 21 bytes can be sent with a theoretical bitrate of 250 kbps (see Fig. 4.3). It becomes clear that this 'overhead' is not negligible when transmitting small packets. Fig. 4.4 shows the energy consumption per transmitted payload byte taking into account the SHR, CCA and the RX-TX turnaround time ('startup overhead').

Secondly, the transmit energy contains for each transmission a part with 'packet overhead', so reducing the number of transmissions will reduce this transmitted 'overhead'. The transmit time is the time it takes to transmit one data packet. This time contains the time it takes to transmit the data payload and to transmit the

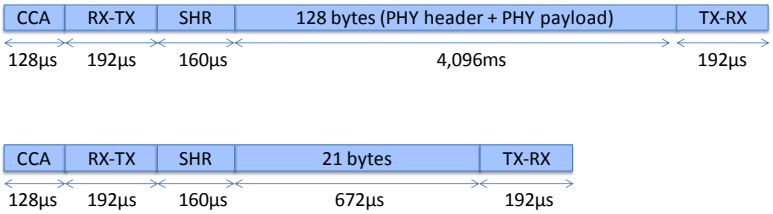


Figure 4.3. Impact of startup overhead on the total transmission time for different packet sizes. It takes the same amount of time to transmit 21 packet bytes and to transmit 1 CCA, 1 SHR and 1 RX-TX turnaround time. Transmitting 128 bytes reduces this negative impact.

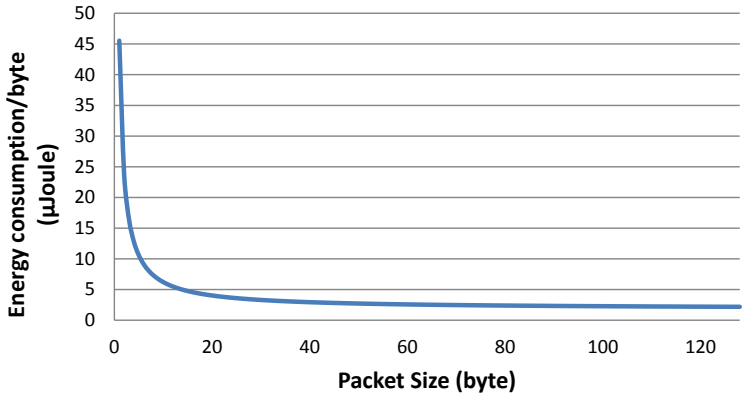


Figure 4.4. Energy consumption per transmitted byte taking into account 1 CCA, 1 SHR and 1 RX-TX turnaround time (CC2420 radio) for different packet sizes

overhead created by protocols. Performing in-network packet aggregation keeps the amount of exchanged payload data, but the overhead introduced by protocols can be reduced. First, an 802.15.4 packet has a physical header of 1 byte. Aggregating 5 packets into 1 can save 4 bytes. When in-network information aggregation is performed, the MAC frame overhead (containing the MAC header and the MAC footer) can be even reduced more. The maximum 802.15.4 MAC frame overhead is 25 bytes (if no security is used), so using in-network information aggregation, this frame overhead should only be sent once instead of for each packet. Furthermore, in 802.15.4, an acknowledgment message can be requested. This ACK message lasts 11 bytes. Finally, it is unlikely that there is no packet error rate, so retransmissions will occur. Fewer transmissions will lead to fewer ACK messages and fewer retransmissions.

We should also note that the total amount of consumed energy also depends

on the used network protocols and their individual protocol settings. For instance, the duty cycle of the MAC protocol will have a big impact on the total energy consumption and some MAC protocols implement the RTS/CTS mechanism, which creates additional overhead per transmission. This study is however out of the scope of this dissertation.

From the above, we can conclude this section with an equation of the amount of transmitted energy that can be saved by applying aggregation.

This transmission energy reduction can be calculated as:

$$E_{tx}^{reduction} = 1 - \frac{E_A}{E_{NA}} \tag{4.2}$$

in which E_A is the total transmission energy consumed when aggregation is performed and E_{NA} is the total energy consumed when no aggregation is performed. Expressing E_A in terms of E_{NA}, this leads to:

$$E_{tx}^{reduction} \approx$$

$$1 - \frac{E_{TO} + E_H^A + DoA_{avg} E_{DATA}^{NA}}{E_{TO} + E_H^{NA} + E_{DATA}^{NA}} \frac{1}{DoA_{avg}} \tag{4.3}$$

in which E_{TO} is the energy consumption caused by transmission overhead per packet transmission. This contains the CCA (Clear Channel Assessment) time, the RX-TX turnaround time but also the overhead created by the MAC protocol, e.g. by beacons. E_H^{NA} is the energy consumed to send a not aggregated packet header, E_H^A is the energy consumed to send a header of an aggregated packet and E_{DATA}^{NA} is the energy consumed to send individual data parts.

4.3.2 Impact on QoS Metrics

The major drawback of in-network aggregation is its negative effect on QoS. The longer is waited to perform aggregation, the higher the delay that these packets experience. Furthermore, when there are no aggregate packets, the queue becomes full and packets will be dropped or sent without aggregating them. These QoS-related trade-offs are discussed in the following section.

4.3.2.1 Impact on Delay

As stated, in-network aggregation has a drawback on the experienced end-to-end delay. Two common ways to perform in-network aggregation are:

- Waiting for the requested degree of aggregation

- Waiting for a specified time-out time

As a consequence, waiting for the requested number of aggregation packets or time-out time will increase the delay with a factor that depends on the degree of aggregation or time-out time and the number of hops between the sender and receiver. The actual delay also depends on the network load. In the best-case scenario, a packet does not have to wait because there are enough packets to be transmitted. This packet can directly be forwarded. This is comparable to a scenario without aggregation. In the worst-case the packet has to wait each time until the time-out timer has expired or until the number of packets to be transmitted equals the degree of aggregation. Combining both approaches can control the maximum time-out time in scenarios with a high degree of aggregation and a low traffic rate.

4.3.2.2 Impact on Reliability

In [14], the authors state that the total amount of delivered information stays the same, but that there is an adverse effect on the variance of this value. The authors have thereby assumed that the link reliability stays the same with and without aggregation, which is however not always true in sensor networks. In CSMA/CA based networks, increasing the degree of aggregation results in fewer exchanged packets, which leads to a lower collision probability and a higher reliability. On the other hand, the authors of [15] have already shown that the collision probability for wireless networks increases when many small packets have to contend with a few large packets. As a consequence, the answer to the question what the impact of in-network aggregation is on the reliability is not as straightforward as initially thought. It mainly depends on the traffic rate, the topology and the used network protocols.

In this paper, we don't focus on link reliability, but on data packet reliability (ratio of the number of data packets received at the destination node and the number of data packets sent) due to queue overflows. To save energy, packets are waiting in the queue until their wait time has passed or when the requested degree of aggregation is reached. However, in networks with many traffic flows to many different (next-hop) destinations, packets can accumulate in the queue waiting endlessly for an appropriate aggregation candidate. When the queue is full, packets will be dropped or will be sent without aggregation. This explains the trade-off between energy efficiency and reliability.

4.3.3 Trade-Off Between Energy Efficiency and QoS Optimizations

There is always a trade-off between energy, delay, reliability and the degree of aggregation. Since there are several conflicting requirements, there is no solution

DoA	Condition	Max Value of X	Bytes used
2	$2 \times (15 + X) <= 118$	44	118
3	$3 \times (15 + X) <= 118$	24	117
4	$4 \times (15 + X) <= 118$	14	116
5	$5 \times (15 + X) <= 118$	8	115
6	$6 \times (15 + X) <= 118$	4	114

Table 4.1. Degree of Aggregation

that is optimal for all objectives. Moreover, an instantaneous optimal solution, if there is one, will change with varying applications and with network conditions.

4.4 In-Network Aggregation in the IoT?

The amount of energy that can be saved by using in-network aggregation depends on the number of packets that are aggregated in a single packet before it is sent to the next-hop node. However, this amount of packets strongly depends on the protocols being used for transferring application data. Highly compact proprietary solutions can be used leading to a high DoA. However, even when using IETF-based IoT protocols (e.g. 6LoWPAN, UDP and CoAP [16]) up to 5 packets can be aggregated as can be seen in Table 4.1 and Fig. 4.5.

In a single 802.15.4 packet, 118 MAC payload bytes are available for doing the aggregation (see also Fig. 4.5). In these 118 bytes, we want to aggregate as many packets as possible, with every aggregated packet having the following composition: 7 (MAC header) + 6 (compressed UDP/IPv6) + 2 (MAC footer) + X (application header + payload, supposing each packet has the same size). It is clear that for an increasing degree of aggregation, the number of bytes available for the application header and payload of every aggregated packet decreases. This is illustrated in Table 4.1.

Using CoAP, a number of bytes from the total of X bytes will be consumed by the CoAP header. The CoAP message format is given in Fig. 4.6.

As the CoAP Message Format already consists of a 4-bytes base binary header, a DoA of 6 is not possible as no bytes are left for the actual payload (see Table 4.1). A DoA of 5 however, leaving 8 bytes for the application header and payload, is realistic. When there are no options available in a CoAP response, which may be assumed for simple sensor network transactions, 3 bytes remain available for the actual payload (as one byte is needed for the one-byte payload marker which indicates the end of options, if present, and the start of the payload). So, for CoAP responses, we may conclude that a DoA of 5 is feasible when using standardized network protocols. When considering a typical CoAP GET request, which does not have a payload, 8 bytes are sufficient for the CoAP header and the encoding of

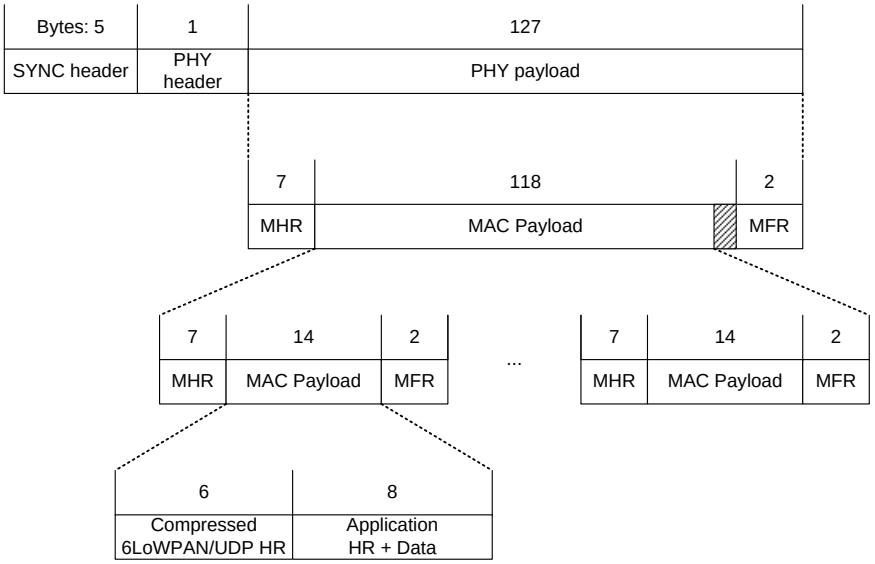


Figure 4.5. Minimal aggregated packet structure for Internet of Things packets based on 802.15.4 packets with SYNC (synchronization) header, PHY (physical) header, MHR (MAC header), MFR (MAC footer) and compressed 6LoWPAN/UDP header for link-local addresses where up to 5 packets can be aggregated.

1 byte			1 byte	2 bytes	TKL bytes	variable	1 byte	variable
V	T	TKL	Code	Message ID	Token (if any)	Options (if any)	OxFF (if payload)	Payload (if any)
2	2	4 bits						

Figure 4.6. CoAP message format

a URI path with length 3. For more complex CoAP packets with more options, we can see from the table that a DoA of 4 is often still possible.

4.5 In-Network Aggregation in QoS Framework

In-network aggregation can be easily implemented in the protocol-independent QoS architecture for wireless sensor networks that was presented in Chapter 2.

The main components used for the in-network aggregation are the *Information Database* and the *QoS Policies Driven Common Queue*. This is shown in Fig. 4.7. In Section 2.4.1, we have defined the Information Database as a database where applications and network protocols register meta data on information parameters. For instance, an application can register the required maximum end-to-end delay and reliability. Not only applications can use this information repository, also

network protocols can use it for storing their control information.

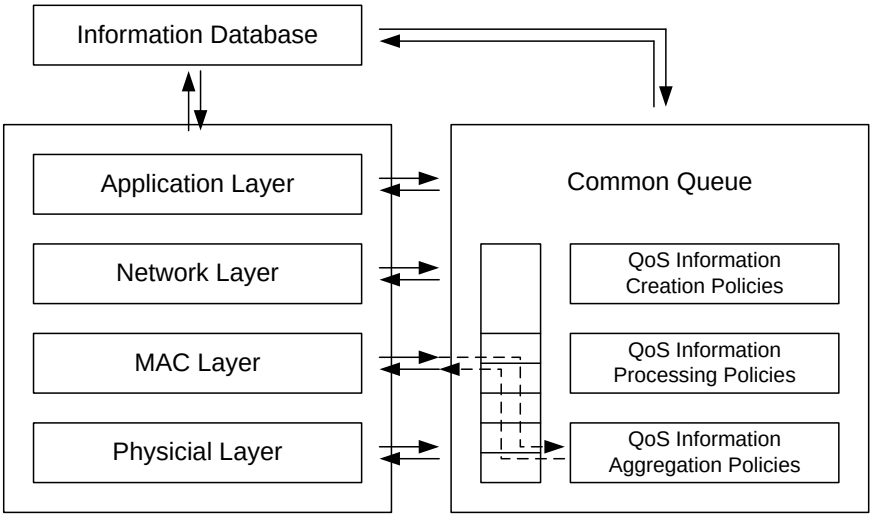


Figure 4.7. Protocol-independent QoS Architecture: components

The second component is the QoS Policies Driven Common Queue. All layers store their packets in a single common queue, while in traditional approaches each layer manages its own packet buffer. We recapitulate the many advantages:

- *A global network overview*. When packets need to be dropped, selection can be made between all the packets in the shared protocol-independent queue, independent of the layer to which they belong. This increases the chance to drop fewer important packets, instead of dropping a more important packet in a certain queue of a certain protocol while fewer important packets are available in other protocol queues.
- *Load-balanced storage*. Temporary high storage requirements for one layer can be balanced with lower requirements of other layers. An optimal overall queue size can be calculated instead of over-provisioning the individual network layer buffers.

4.5.1 Packet Structure

In this dissertation, in-network packet aggregation is performed. So each MAC frame is integrated with its headers into a new MAC frame with a new header. This can be seen in Fig. 4.5. It should be noted that aggregation could also be performed on network level, where each network frame was integrated as MAC payload.

The protocol-independent QoS support is realized by adding the QoS header with a mandatory priority field (Table 2.2) and one or more optional QoS attributes (Table 2.3). Processing and forwarding is based on the priority level, while the attributes are used for additional QoS metrics. In proprietary solutions, the QoS header can be added to the MAC header and lasts between 3 bits and 8 bytes (see Section 2.4.3). For non-proprietary solutions, e.g. when CoAP is used, the QoS header can be added in front of the MAC payload. In this situation, a DoA of 5 is still possible since we have 3 free bytes of MAC payload that can be used for additional QoS header information.

4.6 Conclusion

In-network aggregation is often used in wireless (sensor) networks to reduce energy consumption on sensor nodes with limited battery capacities. In this chapter, we have shown that will this technique is very efficient to reduce energy consumption, it can have a negative impact on QoS metrics such as delay and reliability.

We have also shown that in-network aggregation is still possible with standardized sensor network protocols such as 802.15.4, 6LoWPAN and CoAP. Therefore, we showed how in-network aggregation can be implemented in our QoS framework. In the following chapters, we will show several techniques to improve QoS when in-network aggregation is performed.

References

[1] E. Fasolo, M. Rossi, J. Widmer, and M. Zorzi. *In-network aggregation techniques for wireless sensor networks: a survey*. IEEE Wireless Communications, 14(2):70–87, 2007.

[2] M. E. Oxley and S. N. Thorsen. *Fusion or Integration : What's the difference ?* Transformation, (1).

[3] J. Kulik, W. Heinzelman, and H. Balakrishnan. *Negotiation-based protocols for disseminating information in wireless sensor networks*. Wireless Networks, 8:169–185, March 2002. doi:10.1023/A:1013715909417.

[4] C. Intanagonwiwat, R. Govindan, and D. Estrin. *Directed Diffusion: A scalable and robust communication paradigm for sensor networks*. In Proceedings of MOBICOM, pages 56–67, 2000.

[5] M. Ding, X. Cheng, and G. Xue. *Aggregation tree construction in sensor networks*. In Proceeding of the IEEE 58th Vehicular Technology Conference, volume 4, pages 2168–2172, October 2003. doi:10.1109/VETECF.2003.1285913.

[6] S. Lindsey, C. Raghavendra, and K. Sivalingam. *Data gathering algorithms in sensor networks using energy metrics*. IEEE Transactions on Parallel and Distributed Systems, 13(9):924–935, September 2002. doi:10.1109/TPDS.2002.1036066.

[7] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan. *An application-specific protocol architecture for wireless microsensor networks*. IEEE Transactions on Wireless Communications, 1(4):660–670, October 2002. doi:10.1109/TWC.2002.804190.

[8] A. Manjhi, S. Nath, and P. B. Gibbons. *Tributaries and deltas: efficient and robust aggregation in sensor network streams*. In Proceedings of the 2005 ACM SIGMOD international conference on Management of data, pages 287–298, 2005. doi:10.1145/1066157.1066191.

[9] K. Vaidyanathan, S. Sur, S. Narravula, and P. Sinha. *Data Aggregation techniques in Sensor Networks*.

[10] K. Kalpakis, K. Dasgupta, and P. Namjoshi. *Efficient algorithms for maximum lifetime data gathering and aggregation in wireless sensor networks*. Computer Networks, 42(6):697–716, 2003. doi:10.1016/S1389-1286(03)00212-3.

[11] Zolertia Z1. *Wireless Sensor Node*. http://zolertia.sourceforge.net/wiki/images/e/e8/Z1_RevC_Datasheet.pdf. [Online; accessed 16 January 2014].

[12] Texas Instruments. *CC2420 Radio*. http://www.ti.com/product/cc2420/. [Online; accessed 13 July 2012].

[13] 802.15.4d 2009. *Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs)*.

[14] J. Benson, T. O'Donovan, and J. Sreenan. *Reliability Control for Aggregation in Wireless Sensor Networks*. In Proceedings of the 32nd IEEE Conference on Local Computer Networks, pages 833–840, October 2007. doi:10.1109/LCN.2007.106.

[15] S. H. Nguyen, H. L. Vu, and L. L. H. Andrew. *Packet Size Variability Affects Collisions and Energy Efficiency in WLANs*. In Proceedings of the 2010 IEEE Wireless Communications and Networking Conference, pages 1–6, April 2010. doi:10.1109/WCNC.2010.5506226.

[16] I. Ishaq, D. Carels, G. K. Teklemariam, J. Hoebeke, F. Van den Abeele, E. De Poorter, I. Moerman, and P. Demeester. *IETF Standardization in the Field of the Internet of Things (IoT): A Survey*. Journal of Sensor and Actuator Networks, 2(2):235–287, 2013. doi:10.3390/jsan2020235.

5

Unicast-Based QoS-Aware In-Network Aggregation for Outgoing Wireless Sensor Network Traffic

5.1 Introduction

In this chapter, we focus on the outgoing sensor network traffic, i.e. the traffic that flows from the sensor nodes towards the Internet. This is visualized in Fig. 5.1. This traffic is often multipoint-to-point or point-to-point traffic where one sensor node plays the role of a central sink node. When sensor network traffic has to be routed, the nodes often form a tree topology.

Traditional source-to-sink applications are environmental monitoring and industrial process control. Examples are the drinking water monitoring project in Australia [1] and the WiCon project that focuses on wireless process control in industrial environments [2]. However, emerging applications are home health care / assisted living and smart buildings and cities. An example can be found in the IBBT DEUS project [3] in which elderly people and caretakers in nursing homes are being monitored and tracked.

More and more, these dedicated and sensitive applications are deployed on top of wireless sensor networks as they have lower installation costs: no wires are needed and they can be installed in a plug-and-play fashion. However, these sensor networks are characterized by application-specific devices such as cameras and smartphones and they often exchange data that is subject to more stringent

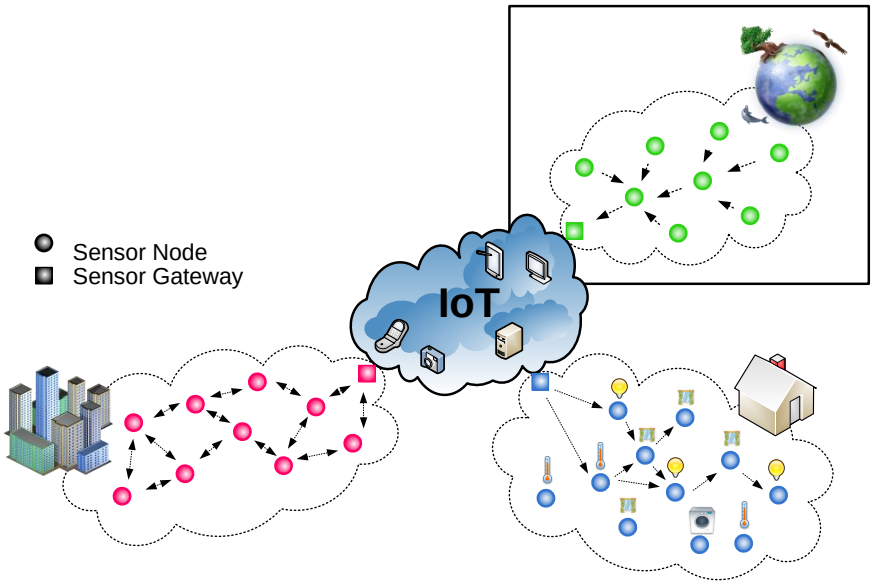


Figure 5.1. Outgoing WSN traffic: traffic flows from the sensor nodes towards the IoT

delay and reliability demands. This imposes new challenges not only on Quality of Service support, but also on energy consumption and maintenance costs due to battery replacement.

When performing in-network aggregation, there is a trade-off between energy consumption and the delivered QoS level. More aggregation will lead to reduced energy consumption, but packets will encounter more delay. In heavily loaded networks, aggregation will reduce the network load and thus increase the overall reliability. However, the impact of packet loss when transmitting aggregated packets can reduce the reliability since multiple information parts are lost at the same time. So there is not only a trade-off between the QoS level and the energy consumption, but also between the different QoS metrics.

In this chapter, a novel protocol-independent QoS-aware in-network aggregation scheme is proposed, which allows making an optimal trade-off between the required QoS level and the energy consumption for next-generation wireless sensor networks. Additionally, this approach allows for an optimal tuning of the number of packets that will be aggregated into one packet (= the degree of aggregation), depending on the load. Moreover, our approach is protocol-independent, making it generic enough to be applied in combination with any existing network protocol.

This chapter proceeds as follows. In Section 5.2, the ongoing related work on multipoint-to-point or point-to-point in-network aggregation is given. Next,

in Section 5.4, the actual QoS-aware in-network aggregation scheme is proposed and in Section 5.5, the simulation results are discussed and analyzed. Section 5.6 concludes this chapter.

5.2 Related Work

The aggregation approaches described in Section 4.2 focus on minimizing the number of packet transmissions and thus reducing energy consumption and extending the network's lifetime. However, since QoS support is important, new in-network aggregation approaches were proposed that also take care of QoS during aggregation. Since this thesis only focuses on lossless aggregation, lossy solutions are not considered. Still, the application and routing layer are allowed to implement some lossy aggregation techniques in order to further reduce the amount of transmitted packets.

In AIDA [4], an adaptive application-independent data aggregation mechanism is presented. This solution contains an aggregation module that resides between the data link and network layer. Aggregation decisions are made in accordance with an adaptive feedback-based packet-scheduling scheme that dynamically controls the degree of aggregation in accordance with the MAC delay. This dynamic feedback scheme is based on the overall queuing delay imposed on AIDA payloads that are waiting for transmission. AIDA cannot differentiate between multiple traffic streams and energy optimization is considered more as a benefit than as a trade-off.

The authors of LUMP [5] propose a simple data aggregation protocol which enables QoS support for applications. Therefore, it prioritizes packets for differentiated services and facilitates aggregation decisions. The architecture has a cross-layer design and is a completely independent module residing between data link and network layer. The priority level represents the tolerable end-to-end latency of the packet. However, this approach does not consider changing traffic load.

Data gathering and aggregation in a distributed, multihop sensor network under specific QoS constraints is investigated and analyzed in Q-DAP & LADCA [6]. Firstly, delay controlled Data Aggregation and Processing (Q-DAP) is performed at the intermediate nodes in a distributed fashion. Each node can decide independently if it performs aggregation. If the delay constraint can be satisfied, the report is deferred for a fixed time interval with a certain probability, otherwise, it is sent to the next hop. If it cannot be satisfied in any case, it is discarded. Secondly, a Localized Adaptive Data Collection and Aggregation (LADCA) approach is proposed for the end nodes. This algorithm defines the data sample rate taking into account energy-efficiency, delay, accuracy and buffer overflow. This solution is layer-dependent (MAC layer for Q-DAP and application layer for LADCA) and is mainly designed for value reporting applications.

In [7], Padmanabh & Vuppala present an adaptive data aggregation algorithm with bursty sources in wireless sensor networks. In this paper, both lossy and lossless aggregation schemes are taken into account. Furthermore, the degree of aggregation is a controllable parameter and buffer management is used to optimize the QoS by minimizing the packet loss due to buffer overflow. This algorithm cannot differentiate between different applications and the energy trade-off is completely ignored.

Akkaya, Younis and Youssef [8] describe an algorithm for achieving maximal possible energy savings through data aggregation while meeting the desired level of timeliness. In order to perform service differentiation and ensure bounded delay for constrained traffic, a weighted fair queuing based mechanism is employed. This approach is protocol-dependent.

5.3 Design Goals

Based on the state-of-the-art research and shortcomings, the main design goals of our QoS-aware in-network aggregation approach are described below.

1. *QoS support: Delay & Reliability*. Multiple QoS objectives such as delay and reliability can be handled at the same time.

2. *Energy-awareness*. Energy consumption is a main concern in sensor networks. Therefore, our solution addresses energy-efficiency by reducing the number of packets being sent.

3. *Protocol-independent aggregation*. Our system is decoupled from the network protocols. This way, it can be used in traditional layered, cross-layer and even in modular or layerless systems with any existing network protocol.

Besides these three main design goals, our approach has several other benefits:

- *Application-independent*. Our approach is generic enough to be used by any application that has one or more QoS requirements in terms of reliability and delay.

- *Information-driven*. The focus is on the exchanged information instead of the exchanged packets. Multiple aggregated information parts can have different QoS requirements. These requirements will be used when making in-network aggregation decisions. Furthermore, also control and meta data information on for instance the remaining energy levels can be taken into account.

- *Multiple heterogeneous applications*. Thanks to the application-independent and information-driven approach, multiple applications with different QoS requirements can be supported at the same time in a single sensor network. Each application with its own QoS requirements will be handled in the optimal way.

- *Self-growing network approach*. The system can adapt itself to changing network, node and application conditions.

Table 5.1 compares these design goals for the different state-of-the-art solutions discussed in Section 5.2.

5.4 Tunable QoS-Aware In-Network Aggregation Scheme

5.4.1 Definition

In this section, we discuss our QoS-aware in-network aggregation scheme. Firstly, we propose a fixed minimum energy aggregation scheme that fulfills the energy requirements. Secondly, a delay control mechanism is added in order to meet the delay constraints. Finally, a reliable load feedback mechanism is added for dealing with reliability constraints. Combining these three schemes leads to the fully tunable QoS-aware in-network aggregation scheme. In Section 5.5, this scheme with its different mechanisms is simulated and the performance is analyzed.

5.4.1.1 Fixed Minimum Energy Aggregation Scheme

From Section 4.3.1, it is clear that the fewer packets are sent, the more energy is saved. Therefore, this scheme uses a maximum degree of aggregation (DoA) value. Aggregation now can take place as long as the remaining MAC payload size is bigger than one MAC frame. The maximum degree of aggregation will depend on the length of the exchanged data packets. The drawback of this scheme is that packets can wait endlessly on the required number of aggregation candidates. To overcome this issue, we can introduce a time-out time or wait time (= the maximum time that a packet is allowed to wait in one node before it should be sent). In current research solutions, this value is often arbitrarily chosen and not tuned to the instantaneous network and application conditions. However, we will introduce below the delay control mechanism which allows us to use an optimal and well-defined wait time value.

Design Goals	AIDA	Lump	Q-DAP & LADCA	Padma. & Vuppala	Akkaya, Younis & Youssef	Ours
Delay & Reliability	✗	✗	✗	✗	✗	✓
Energy-awareness	✗	✗	✓	✗	✓	✓
Protocol-independent	✓	✓	✗	✓	✗	✓
Heterogeneous applications	✗	✓	✗	✗	✗	✓
Application-independent	✓	✓	✗	✓	✓	✓
Information-driven	✗	✓	✓	✓	✓	✓
Self-growing	✓	✗	✓	✓	✓	✓

Table 5.1. Comparison of the state-of-the-art QoS-aware in-network aggregation solutions and our solution in terms of QoS-related design goals

5.4.1.2 Delay Control Mechanism

The delay control mechanism uses the previously introduced priority levels and delay attributes. Initially, each packet receives a priority level, based on Table 2.2. When no delay attributes are added, the maximum degree of aggregation level is applied (see fixed minimum energy aggregation scheme in Section 5.4.1.1). However, if additional information is available on the maximum end-to-end delay or the maximum per-hop delay (see Table 2.3), this information is used for faster aggregation. In case the maximum end-to-end delay is available, the maximum wait time can be calculated by dividing this value by the number of hops along the path to the destination. This value can dynamically be recalculated along the routing path if the topology or the route changes.

Aggregation is then performed:

- When the maximum degree of aggregation is reached or
- When the maximum wait time is reached.

In the last case, as many as possible packets are aggregated when this wait time deadline is reached.

5.4.1.3 Reliable Load Feedback Mechanism

In networks with different sink nodes, it is possible that the queue becomes fully occupied with several packets with different (next-hop) destinations. In this situation, it is possible that the queue becomes full when both the maximum wait time and the maximum degree of aggregation are not yet reached. As a consequence, packets will be dropped, as shown in Fig. 5.2. To overcome this issue, the load feedback mechanism is applied. If the queue load increases, the degree of aggregation is adjusted to a level that allows faster aggregation and reduces the probability that a packet is dropped from the queue. When the number of remaining free spaces in the queue is lower than the degree of aggregation, the number of packets for each (next-hop) destination is calculated, and the maximum number is selected as the new degree of aggregation. This calculation is performed again for each aggregation round as long as the remaining free spaces in the queue are lower than the degree of aggregation.

Combining the three approaches leads to a fully tunable QoS-aware in-network aggregation scheme.

5.4.2 Operational Working

In this section, the operational working of the tunable QoS-aware in-network aggregation scheme is shown. A distinction is made between the initialization phase,

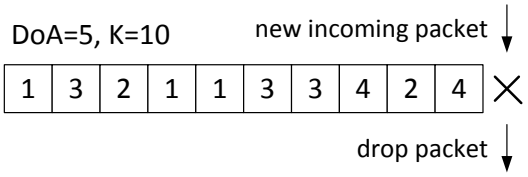


Figure 5.2. High traffic load with many different destinations: packet drops will occur

the pre-aggregation rules and the actual aggregation rules. At the end, the complete algorithm is shown.

5.4.2.1 Initialization

In the initialization phase, the initial maximum degree of aggregation and, for each application, the specific QoS requirements in terms of priority, delay and reliability, are set (see Tables 2.2 and 2.3).

5.4.2.2 Pre-Aggregation Rules

It is possible that the queue is full when a new packet arrives. Instead of dropping that new packet, an existing packet may be dropped. Typically, this will be a packet which deadline has almost been reached, or a packet with a lower priority.

5.4.2.3 Aggregation Rules

Aggregation is performed by aggregating multiple MAC packets into a new MAC packet with a new MAC header. The following variables are used:

- The number of packets for the same next-hop: $\boldsymbol{n_d}$.
- The maximum degree of aggregation: $\boldsymbol{a}$.
- The maximum wait time of packet $\boldsymbol{x}$ in one node: $\boldsymbol{T_x^{max}}$.

The following three QoS-aware in-network aggregation rules are defined:

- **Rule 1**. When $\boldsymbol{n_d \geq a}$, the maximum number of aggregated packets for the same next-hop is reached, and the aggregated packet will be sent to the next-hop node.
- **Rule 2**. When $\boldsymbol{n_d < a}$ and $\boldsymbol{\exists x \in S_d : T_x^{max} - T_x < \epsilon}$, the packet x for which the deadline has almost been reached will be aggregated with as many other packets as possible that have the same next-hop, highest priority packets first, and will be sent to the next-hop node. $\boldsymbol{S_d}$ is the set of packets

with the same next-hop $\boldsymbol{d}$, $\boldsymbol{T_x}$ is the current wait time in the node of packet x and $\boldsymbol{\epsilon}$ is a certain threshold.

- **Rule 3**. The maximum degree of aggregation $\boldsymbol{a}$ is tuned based on the remaining free queue size and the amount of packets for a certain next-hop node.

An overview of the algorithm described above is given in Algorithm 1.

$C \leftarrow$ Common Queue
$M \leftarrow$ list of all MAC packets ready for sending, sorted on highest priority first
$A \leftarrow \{\}$ (empty list to hold packets that can be aggregated)
DoA = maximum Degree of Aggregation
$h_a \leftarrow$ additional MAC header with aggregation info
if C.freespaces $= 0$ **then**
    Drop packet with lowest priority
end if
for all MAC packet $m \in M$ **do**
    $A \leftarrow A \cup m$
    $D \leftarrow$ list of all packets with same next-hop as m, sorted on highest priority first
    for all $d \in D$ **do**
        if byteLength $(A + d + h_a) <$ max MAC packet byteLength **then**
            $A \leftarrow A \cup d$
        end if
    end for
    if $\#A =$ DoA **then**
        Aggregate
        return Aggregated packet
    else if maxNodeWaitTime(m) reached **then**
        Aggregate
        return Aggregated packet
    end if
    $A \leftarrow \{\}$
end for
if C.freespaces $<$ DoA **then**
    Lower DoA to the instantaneous maximum number of packets for the same (next-hop) destination and aggregate
    return Aggregated packet
end if

Algorithm 1. Tunable QoS-aware In-Network Aggregation Scheme

Parameter	Value
Simulation time	2700 s
Simulation field	75 m x 75 m
Number of nodes	256
Node deployment	16x16 grid
Routing protocol	DYMO / CTP
MAC protocol	tunable CSMA based MAC (Castalia) with duty cycle 0.2 (100 ms sleep/400 ms listening)
Radio	CC2420 with no transmission errors
Wireless channel	no interference
Collision domain	$\pm$ 16 m, depends on the receiver signal strength ($\pm$ 2 hops)
Queue size (K)	15
DoA	5

Table 5.2. Outgoing WSN traffic: simulation settings

5.5 Simulation Results

As explained in section 3.3.1, simulations are done using Castalia. In the remainder of this chapter, we use a scenario with 256 nodes, deployed in a square grid with size 75 m. This setup was chosen in order to meet the minimal requirements of RFC 5826 for home automation routing in low power and lossy networks [9].

In our evaluation, we analyze 1 energy metric: energy consumption and 3 QoS metrics: end-to-end packet delay, per-node packet delay and packet reliability. A packet is reliable received when it is received without errors at the destination node. Evaluation is done for 30 different traffic rates with the simulation settings given in Table 5.2.

For the fixed minimum energy aggregation scheme and the delay control mechanism, we use a multipoint-to-multipoint application in which 20 out of 256 nodes are transmitting packets. For CTP routing, 20 out of 256 nodes are transmitting packets to one fixed sink node (node 0), while for DYMO routing, 20 out of 256 nodes are transmitting packets to two fixed sink nodes (node 6 and node 243). These topologies are given in Fig. 5.3. The nodes are chosen randomly and each transmitting node generates application packets with an average traffic rate as given on the x-axis of the figures of the simulation results. The common queue can contain 15 packets in both scenarios. These 15 packets are a realistic assumption. For instance, the Zolertia Z1 sensor node [10] has 8 kB of RAM, but 4-6 kB is often used for code, which leaves only 2-4 kB for buffering. In the scenario in which the reliable load feedback mechanism is investigated, all nodes except 20 randomly chosen destination nodes are sending packets.

A simulation lasts 3600 seconds, but statistics are generated in steady state

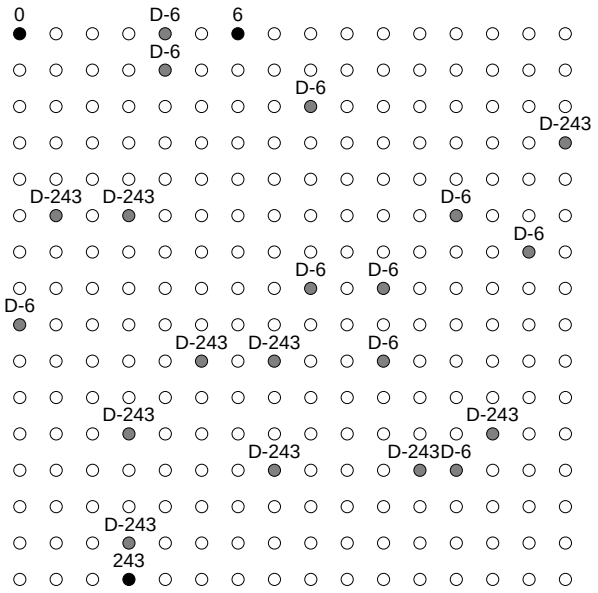


Figure 5.3. Network Topology. The black nodes are the destination nodes: node 0 is the destination node for CTP and nodes 6 and 243 are the sink nodes for DYMO. The gray nodes are the sending nodes (to node 0 for CTP, and to the destination given in the figure for DYMO (node 6 or node 243))

between 300 and 3000 seconds. After 100 seconds, the network is up and running and the DYMO routes remain active during the entire simulation, as can be seen in Fig. 5.4. This is for instance the case when there is a fixed communication session between the sensor and actuator with regular traffic. Simulation statistics later than 3000 seconds are not counted to ensure that all generated packets can reach their destination within the evaluated time. Furthermore, simulations are performed for different average traffic rates (from 1 to 30 packets/min) as given on the x-axis of the figures. For instance, when the average traffic rate is 5 packets/min, nodes choose a random traffic rate between 1 and 10 packets/min. A higher average packet rate will lead to a higher network load. These traffic rates can be justified by a CoAP scenario where a message will be sent every time the max-age expires. This value is default 1 packet/min. When values change more frequently, more messages will be sent. The maximum packet timeout was set to 10 seconds. This value gives nodes at low traffic rates enough time to find aggregate candidates before their timeout time passes.

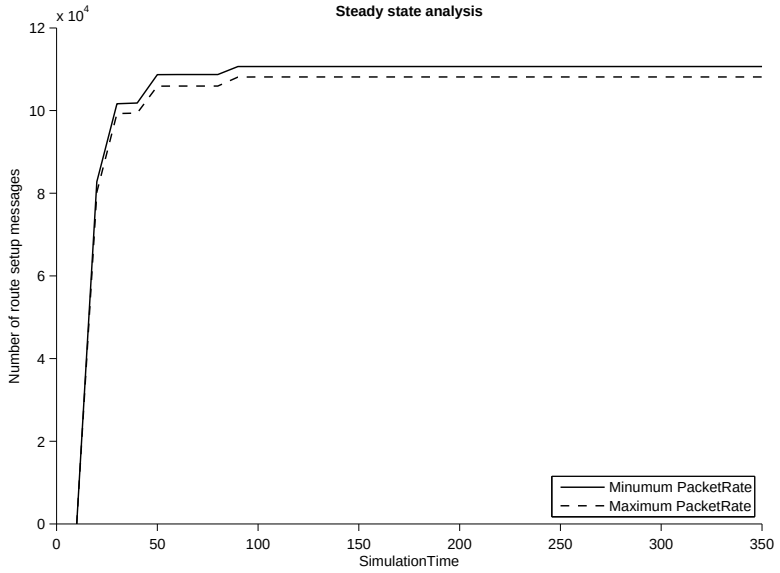


Figure 5.4. Steady State Calculation

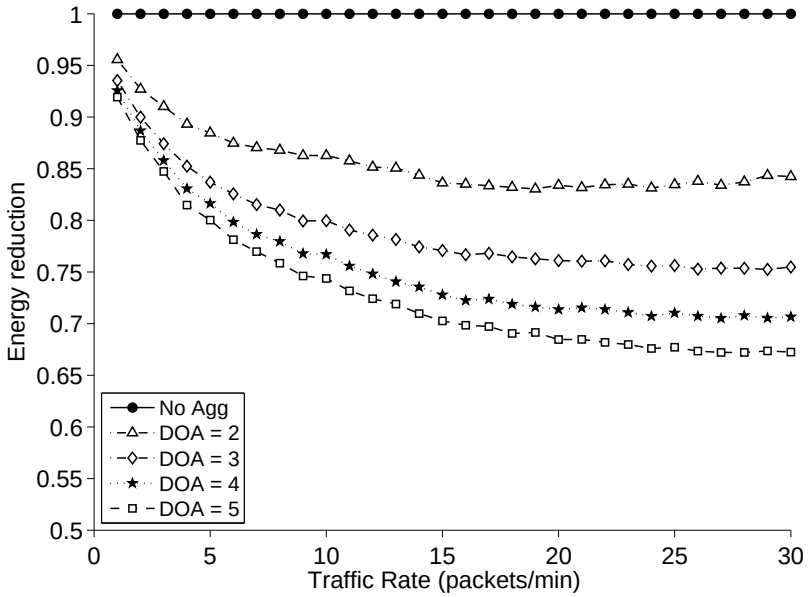


Figure 5.5. Fixed Minimum Energy Scheme: energy reduction with CTP routing protocol

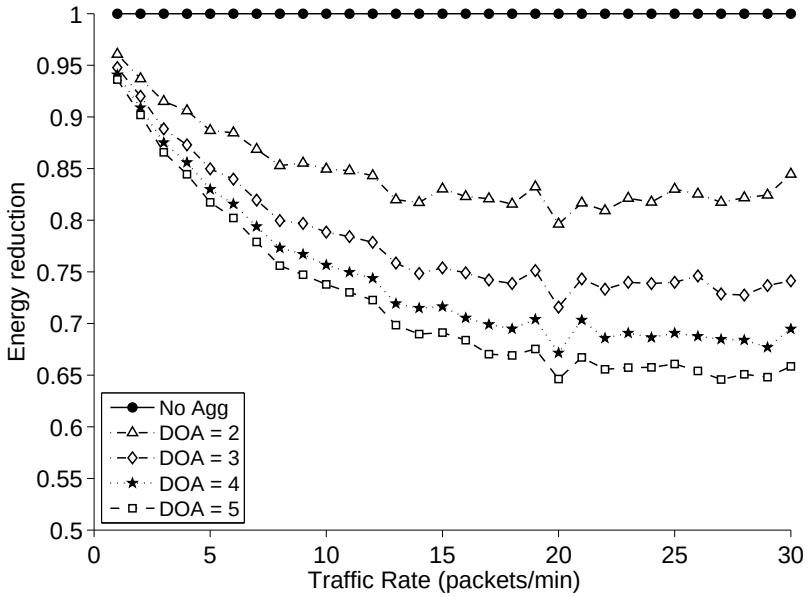


Figure 5.6. Fixed Minimum Energy Scheme: energy reduction with DYMO routing protocol

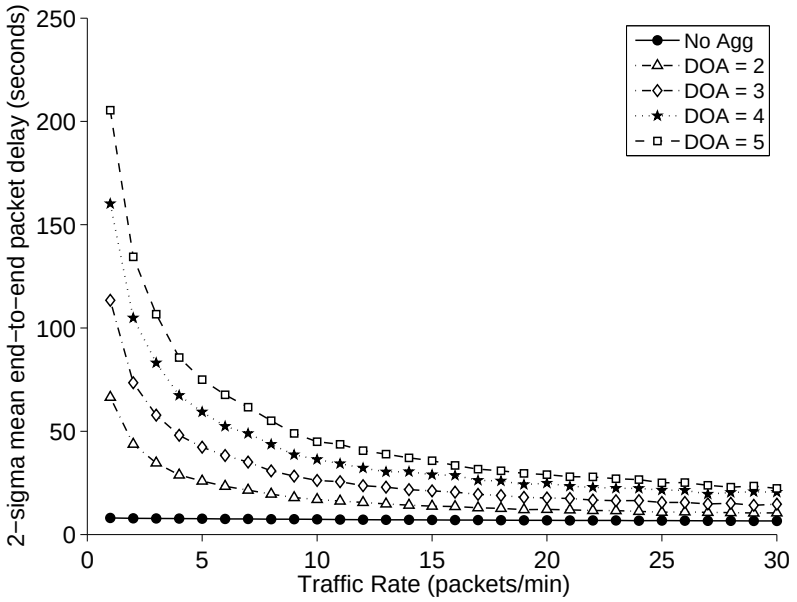


Figure 5.7. Fixed Minimum Energy Scheme: end-to-end delay with CTP routing protocol

5.5.1 Fixed Minimum Energy Aggregation Scheme

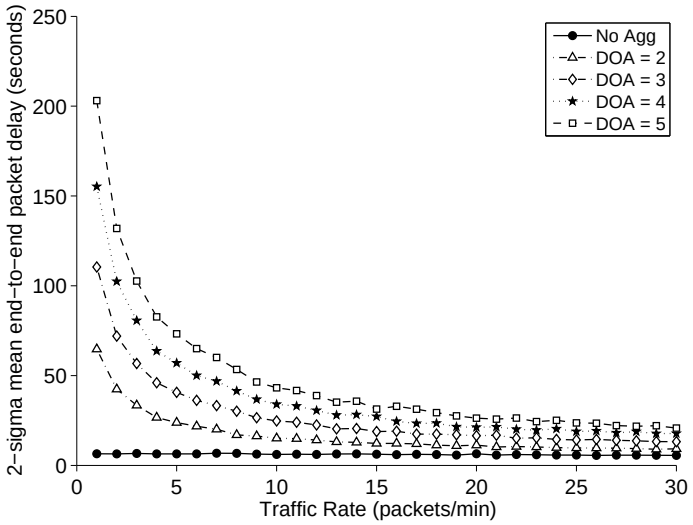


Figure 5.8. Fixed Minimum Energy Scheme: end-to-end delay with DYMO routing protocol

In Figs. 5.5 and 5.6, the energy reduction for different degrees of aggregation (where DOA=X stands for aggregating X packets into a new packet) compared to the no aggregation scenario is given for the fixed minimum energy aggregation scheme for CTP routing and DYMO routing respectively. The diagrams show that when the traffic rate increases, up to 35 % of energy can be saved.

Figs. 5.7 and 5.8 show the impact of this mechanism on the end-to-end delay for CTP routing and DYMO routing respectively and Figs. 5.9 and 5.10 show the respective impact on reliability. These figures show that the fixed minimum energy aggregation scheme leads to a higher end-to-end delay, which is normal since packets will have to wait until there are enough packets to aggregate with. This effect has the most impact when the traffic rate is low, since packets have to wait a long time before they get aggregated. When the traffic rate increases, aggregate candidates become available more quickly so that aggregation can be performed earlier. From the reliability figures, aggregation leads to an improved reliability since fewer packets are in the air and fewer transmissions will fail. We can see that for CTP the reliability for no aggregation is lower than for DYMO. The reason can be found in the fact that with CTP routing, all the packets are routed towards the same sink node, and at some intermediate nodes, much traffic comes together. At these points, there is a higher chance that transmissions will fail. Since the traffic with DYMO is more distributed along the topology, this effect is less distinct.

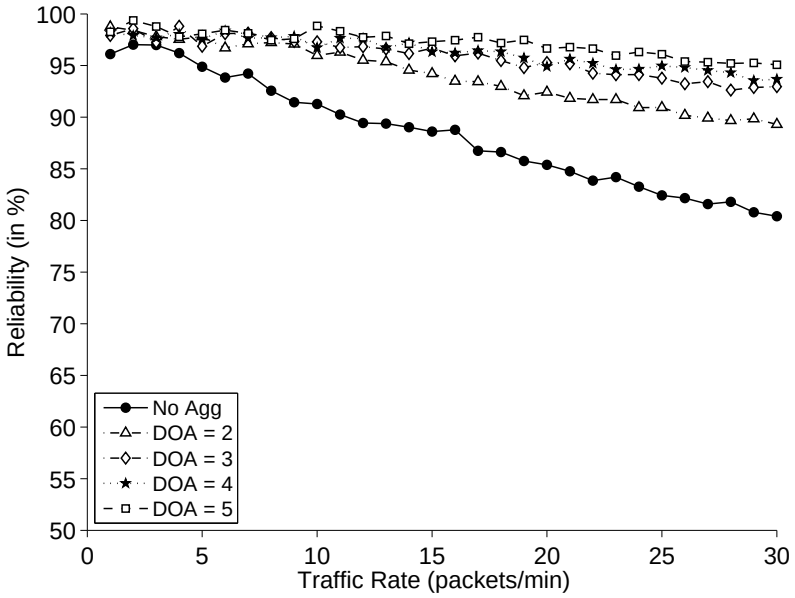


Figure 5.9. Fixed Minimum Energy Scheme: reliability with CTP routing protocol

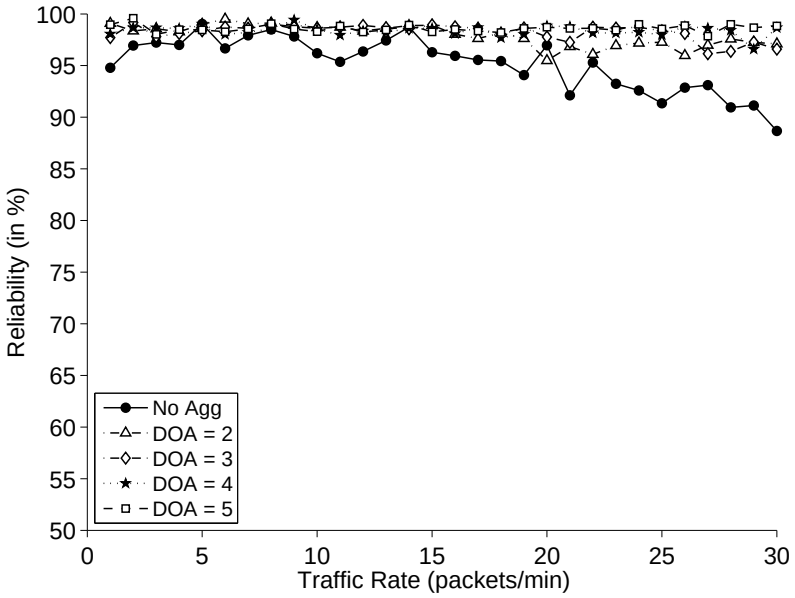


Figure 5.10. Fixed Minimum Energy Scheme: reliability with DYMO routing protocol

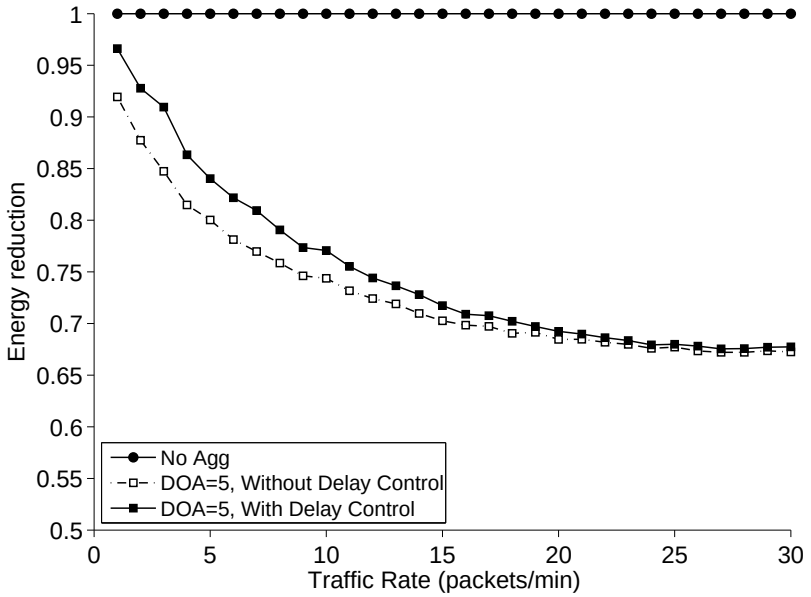


Figure 5.11. Fixed Minimum Energy + Delay Control Mechanism: energy reduction with CTP routing protocol

5.5.2 Delay Control Mechanism

In Figs. 5.11 through 5.18, the impact of the fixed minimum energy aggregation scheme with the delay control mechanism for CTP and DYMO respectively is shown. Simulations are done for the reference scenario in which no aggregation is performed and for the scenario with $DoA = 5$, with and without the delay control mechanism. To simplify simulations and analysis, the maximum per-node delay (maximum node wait time) was defined to be 10 seconds for each node. Figs. 5.11 and 5.12 show that more energy is consumed when the delay control mechanism is applied. This is as expected since packets are already aggregated after that their maximum wait time has reached instead of when there are enough packets, so there are more transmissions with fewer aggregated packets in it, which will increase the energy consumption. However, Figs. 5.13 through 5.16 show that the per-node delay at low traffic rates stays under the required 10 seconds which was the intention of the delay control mechanism. Figs. 5.13 and 5.14 show that starting from an average traffic rate of 4 packets/min, the end-to-end delay with the delay control mechanism becomes bigger than without applying the delay control mechanism. The reason can be found in the fact that aggregation is performed earlier with the delay control mechanism. In this situation, aggregation occurs for instance when there are 4 instead of 5 packets in the queue because the 10 seconds

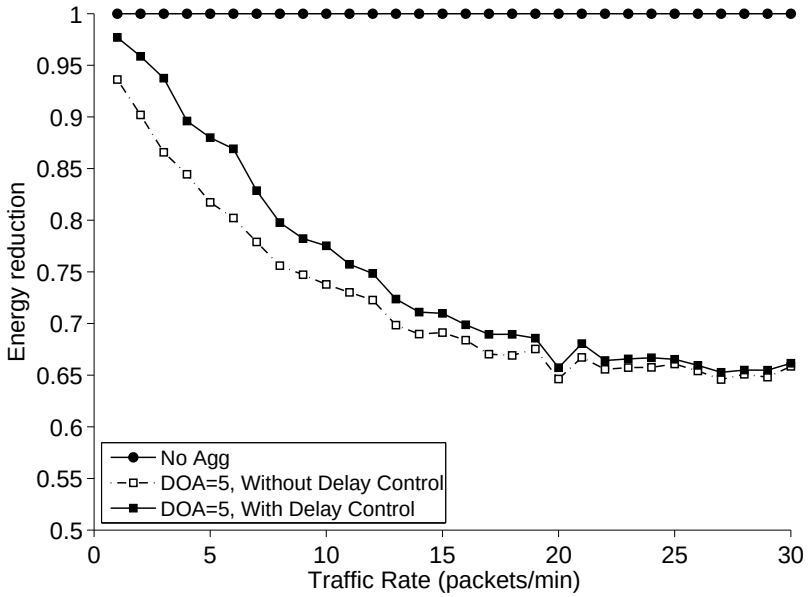


Figure 5.12. Fixed Minimum Energy + Delay Control Mechanism: energy reduction with DYMO routing protocol

time-out is reached. As a consequence, in the next-hop node, there will arrive 4 aggregated packets, and when no other packets enters that node, they will wait again 10 seconds before being aggregated and sent to the next-hop node. This was not the case if 5 packets were aggregated, since they are immediately sent to the next-hop node. This is the price we pay for guaranteeing a maximum per-hop delay of maximum 10 seconds, as shown in Figs. 5.15 and 5.16. Figs. 5.17 and 5.18 show that the reliability with and without delay control mechanism is almost the same, outliers excluded.

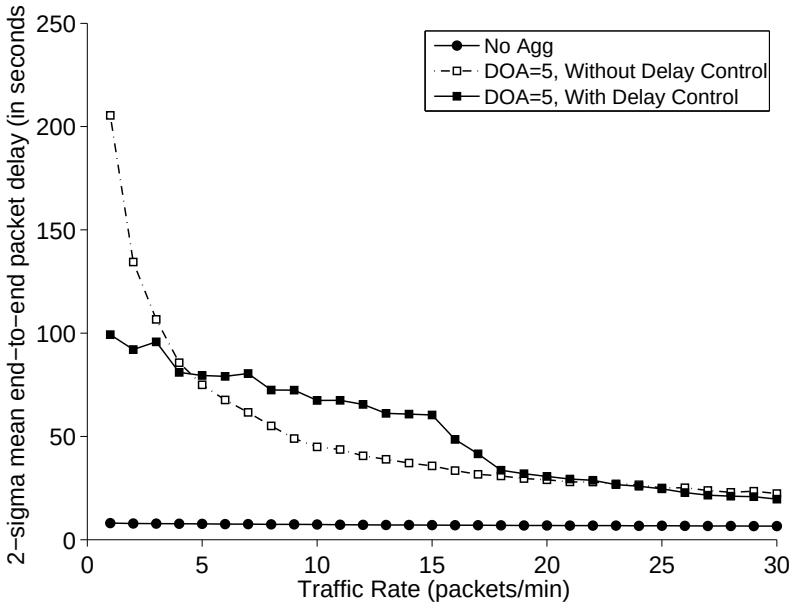


Figure 5.13. Fixed Minimum Energy + Delay Control Mechanism: end-to-end delay with CTP routing protocol

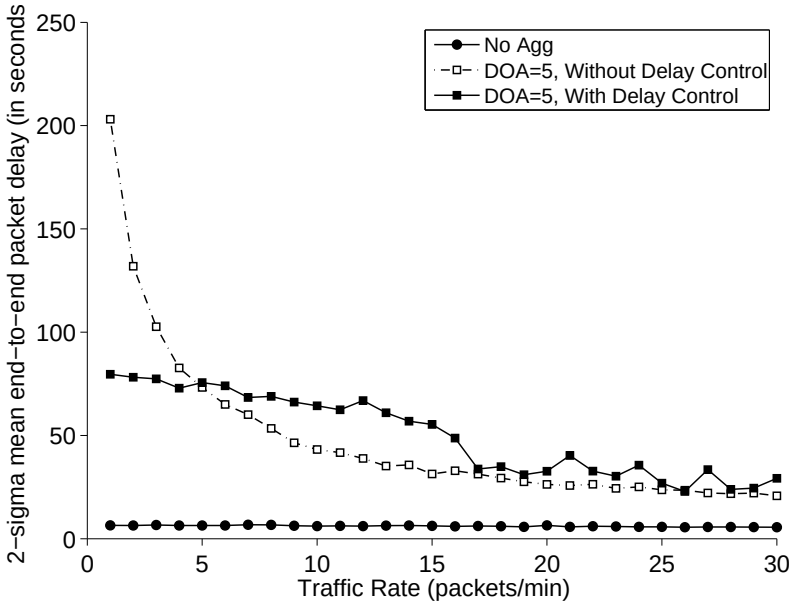


Figure 5.14. Fixed Minimum Energy + Delay Control Mechanism: end-to-end delay with DYMO routing protocol

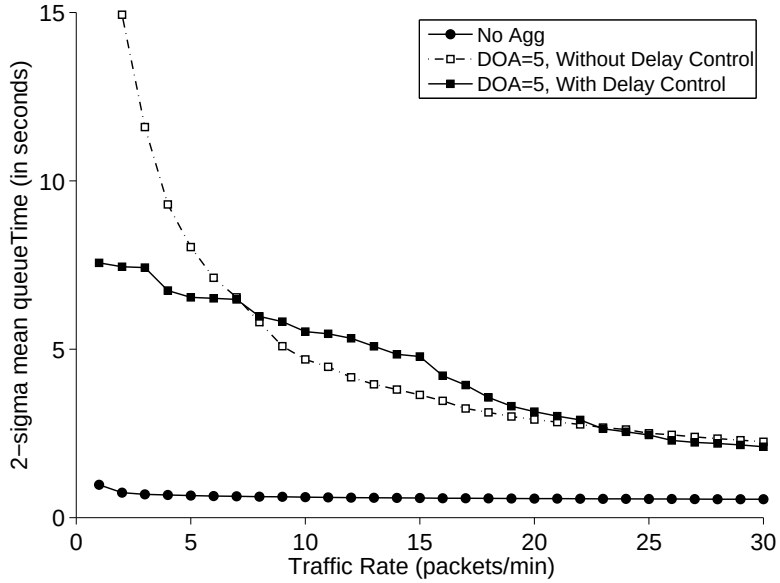


Figure 5.15. Fixed Minimum Energy + Delay Control Mechanism: per-node delay with CTP routing protocol

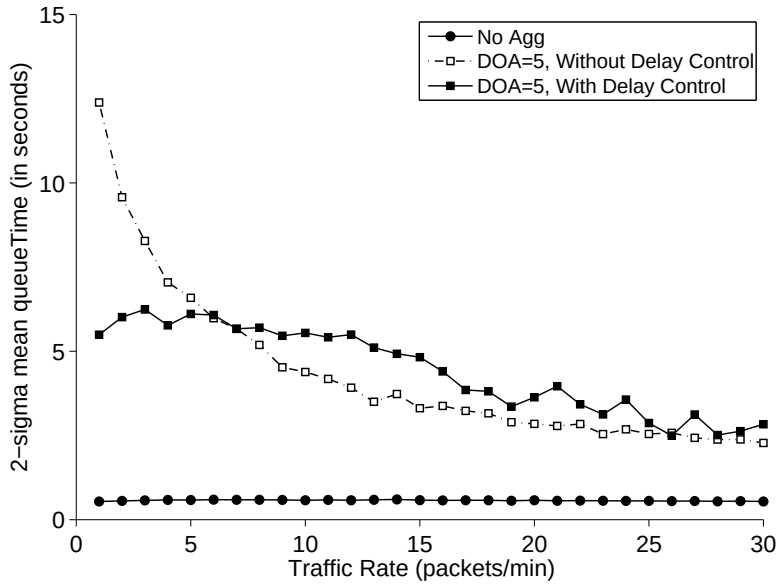


Figure 5.16. Fixed Minimum Energy + Delay Control Mechanism: per-node delay with DYMO routing protocol

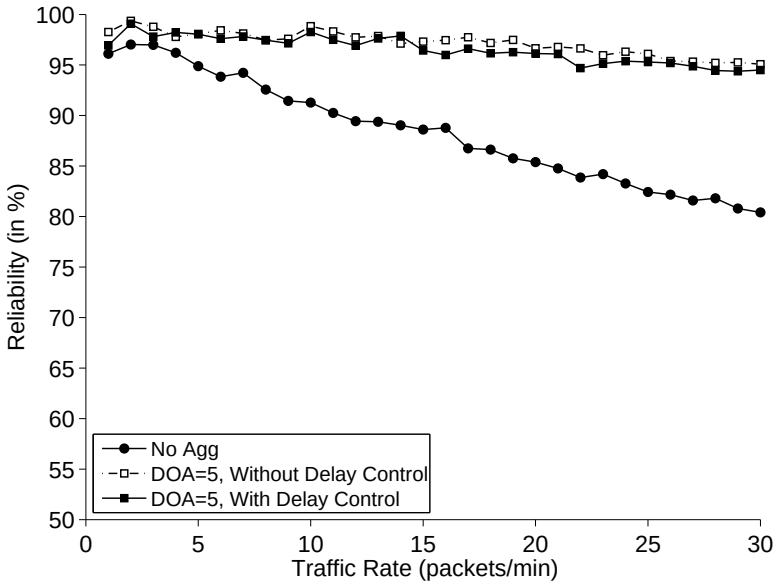


Figure 5.17. Fixed Minimum Energy + Delay Control Mechanism: reliability with CTP routing protocol

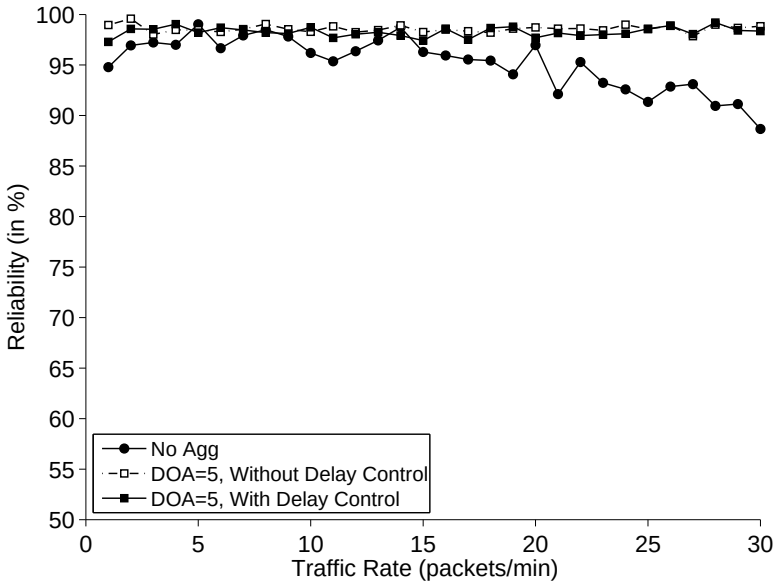


Figure 5.18. Fixed Minimum Energy + Delay Control Mechanism: reliability with DYMO routing protocol

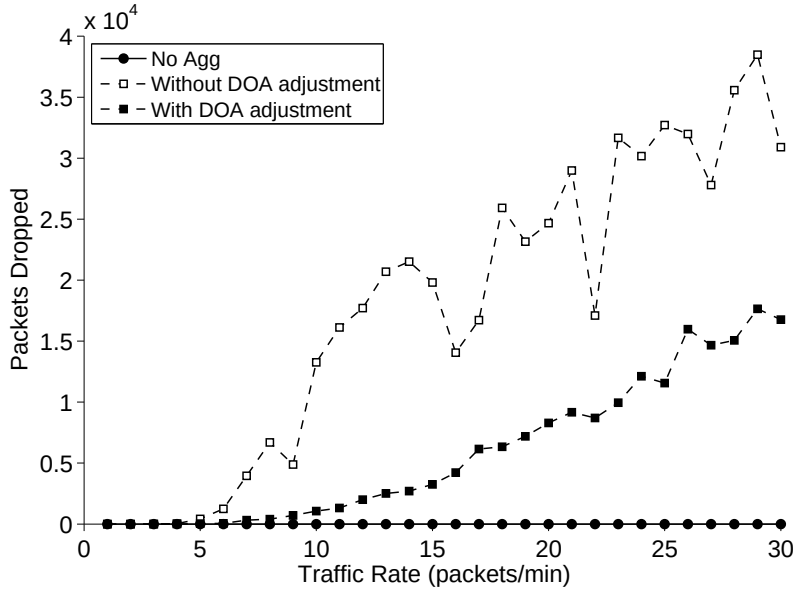


Figure 5.19. Fixed Minimum Energy + Delay Control + Reliable Load-Feedback Mechanism: number of dropped packets with DYMO routing protocol

5.5.3 Reliable Load Feedback Mechanism

In Figs. 5.19 through 5.22, the impact of the fixed minimum energy with both the delay control mechanism and the reliable load feedback mechanism is displayed. These simulations were only performed for the DYMO routing protocol, since only with DYMO, packets are routed to different destinations. Instead of 20 out of 256 nodes that transmit packets, all nodes except the 20 destination nodes are transmitting packets.

The figures display that adjusting the degree of aggregation significantly reduces the number of dropped packets in the queue (Fig. 5.19), which improves the overall reliability (Fig. 5.20) up to 14 % and reduces the end-to-end packet delay up to 18 % (Fig. 5.21). Indeed, faster aggregation leads to less delay. Fig. 5.22 shows that these improvements come at the cost of somewhat higher energy up to 3%. Comparing with the scenario where no aggregation is performed, the fully tunable QoS-aware in-network aggregation scheme leads to a reliability increase up to 16 %, but there is already a performance increase since the beginning, the delay is however increased, but still within the limits and with an energy reduction up to 19 %

The above simulations were only performed for one simulation seed. However, to be statistically valid, simulations need to be done for different seeds. To meet

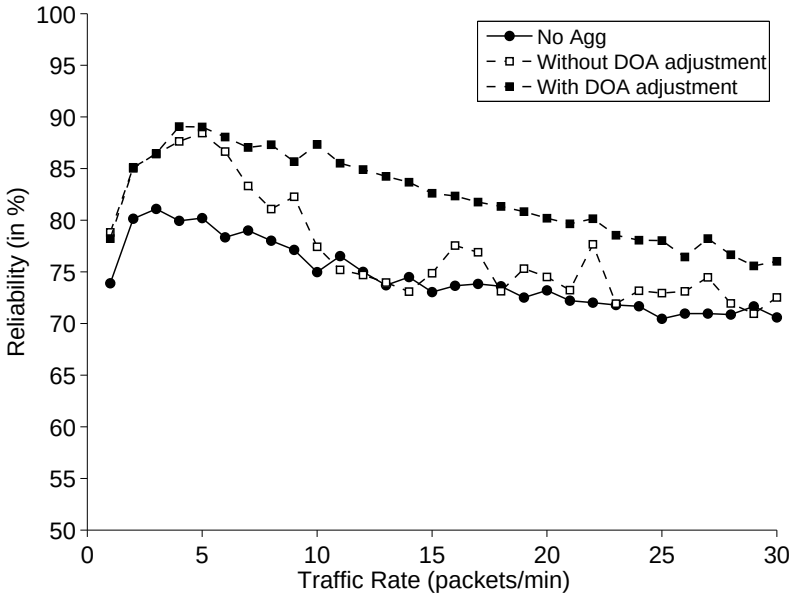


Figure 5.20. Fixed Minimum Energy + Delay Control + Reliable Load-Feedback Mechanism: reliability with DYMO routing protocol

this requirement, we performed several simulations for the reliable load feedback scheme, since this scheme contains the whole tunable QoS-aware in-network aggregation scheme. Figs. 5.23 through 5.26 show that the results discussed above are following the general trends.

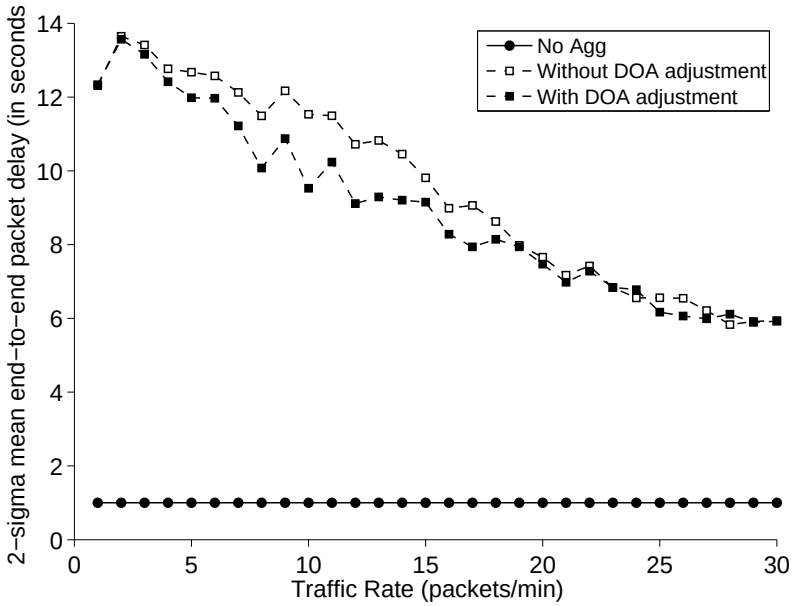


Figure 5.21. Fixed Minimum Energy + Delay Control + Reliable Load-Feedback Mechanism: end-to-end delay with DYMO routing protocol

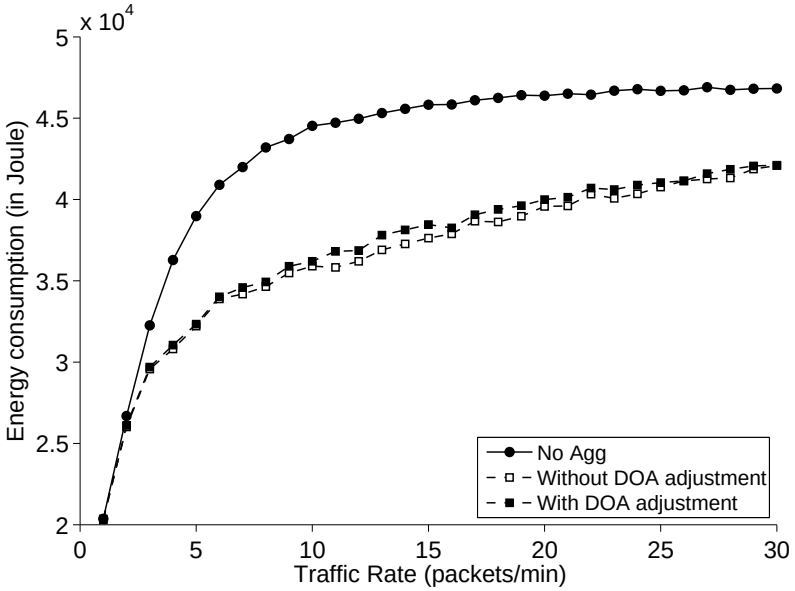
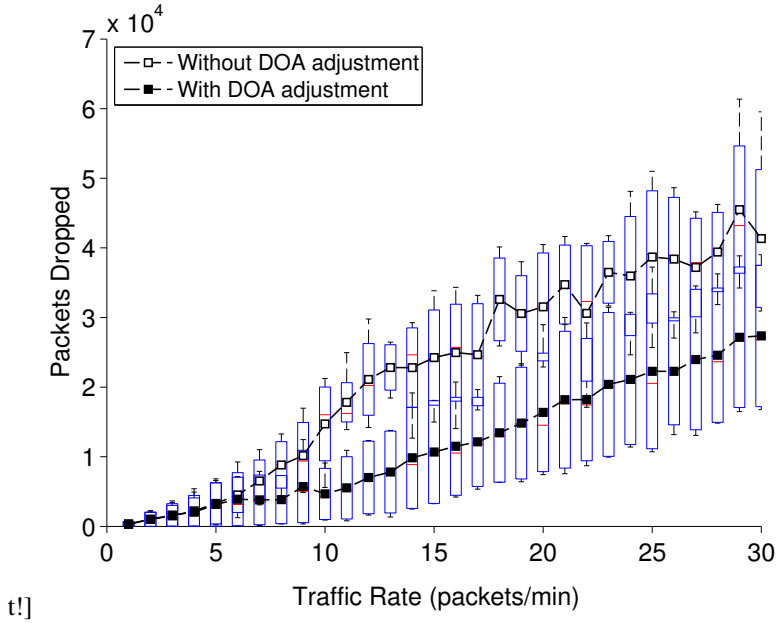


Figure 5.22. Fixed Minimum Energy + Delay Control + Reliable Load-Feedback Mechanism: energy consumption with DYMO routing protocol



t!]

Figure 5.23. Fixed Minimum Energy + Delay Control + Reliable Load-Feedback Mechanism: number of dropped packets with DYMO routing protocol

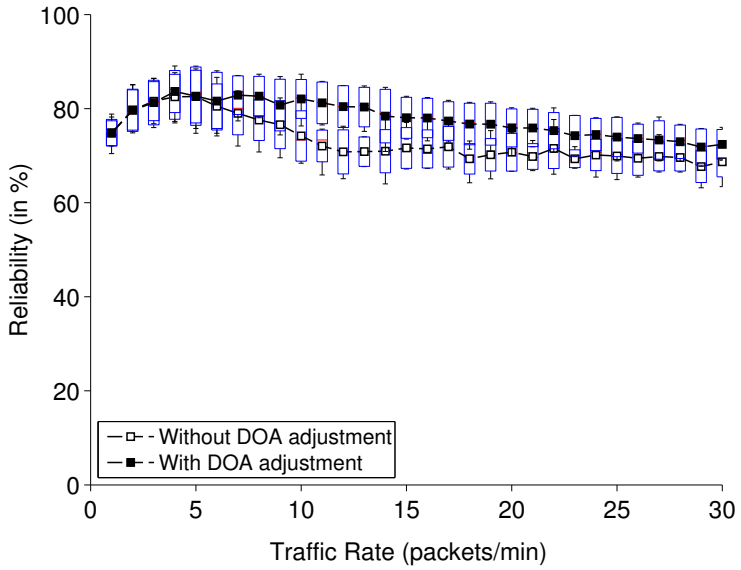


Figure 5.24. Fixed Minimum Energy + Delay Control + Reliable Load-Feedback Mechanism: reliability with DYMO routing protocol

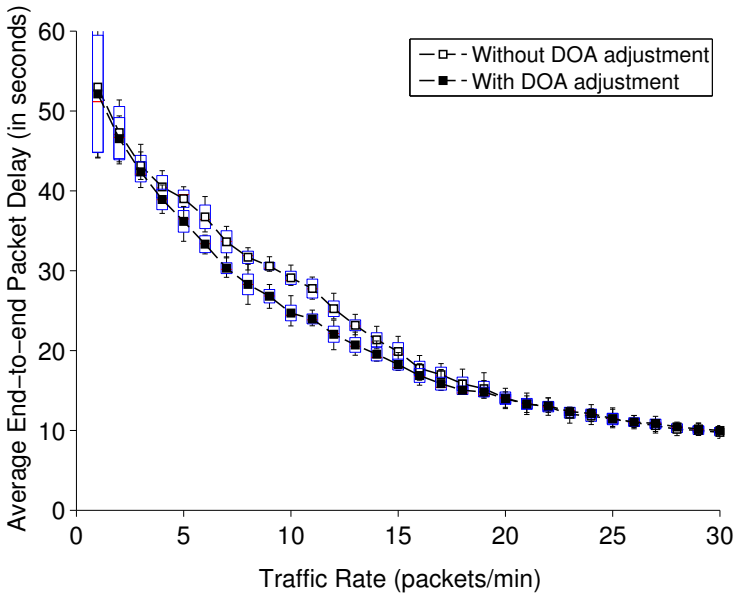


Figure 5.25. Fixed Minimum Energy + Delay Control + Reliable Load-Feedback Mechanism: end-to-end delay with DYMO routing protocol

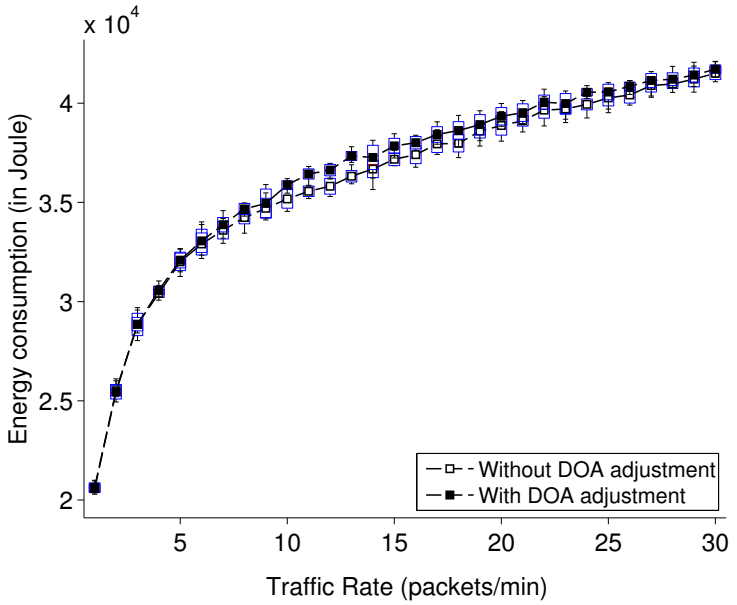


Figure 5.26. Fixed Minimum Energy + Delay Control + Reliable Load-Feedback Mechanism: energy consumption with DYMO routing protocol

5.6 Conclusion

In this chapter, we focused on in-network aggregation for sensor network traffic that is routed from different source nodes towards one or more sink nodes.

We propose a QoS-aware in-network aggregation scheme that is able to reduce energy consumption while still delivering the requested Quality of Service needs. Our solution is protocol-independent, lightweight, adaptive and modular. It can be easily implemented in layered and layerless architectures with any existing network protocols.

We show that the energy consumption can be reduced up to 35 % while still meeting the delay and reliability requirements by applying the minimum energy aggregation scheme and the delay control mechanism. In networks with different sink nodes, applying the reliable load feedback mechanism leads to an increase in reliability of up to 14 % for high traffic rates and a delay reduction up to 18 %, but at the cost of slightly higher energy consumption (up to 3%) compared to the delay control mechanism. We have also compared the fully tunable QoS-aware in-network aggregation scheme with the scenario in which no aggregation is applied. We have shown that the fully tunable QoS-aware in-network aggregation scheme leads to an overall reliability increase up to 16 %. The delay is however increased, but still within the given limits and there is an overall energy reduction up to 19 %.

References

[1] CSIRO and a local water authority. *Smart sensors monitoring water quality and catchment health*. http://www.csiro.au/Outcomes/Water/Water-information-systems/smart-sensors-monitoring-water-quality.aspx. [Online; accessed 13 July 2012].

[2] ABB, DNV, Kongsberg Maritime, SINTEF, and Statoil. *The WiCon project*. http://www.wiconproject.com/. [Online; accessed 13 July 2012].

[3] DEUS. *Deployment and Easy Use of wireless Services*. http://ilabt.ibbt.be/. [Online; accessed 13 July 2012].

[4] T. He, B. M. Blum, J. A. Stankovic, and T. Abdelzaher. *AIDA: Adaptive Application-Independent Data Aggregation in Wireless Sensor Networks*. ACM Transactions on Embedded Computing System, Special issue on Dynamically Adaptable Embedded Systems, 3(2):426–457, 2004.

[5] J. Jeong, J. Kim, W. Cha, H. Kim, S. Kim, and P. Mah. *A QoS-aware data aggregation in wireless sensor networks*. In Proceedings of the 12th International Conference onA dvanced Communication Technology, pages 156–161, February 2010.

[6] J. Zhu, S. Papavassiliou, and J. Yang. *Adaptive Localized QoS-Constrained Data Aggregation and Processing in Distributed Sensor Networks*. IEEE Transactions on Parallel and Distributed Systems, 17:923–933, 2006.

[7] K. Padmanabh. *An Adaptive Data Aggregation Algorithm in Wireless Sensor Network with Bursty Source*. Wireless Sensor Network, 1(3):222–232, 2009.

[8] K. Akkaya, M. Younis, and M. Youssef. *Efficient Aggregation of Delay-Constrained Data in Wireless Sensor Networks*. In Proceedings of the 3rd ACS/IEEE 2005 International Conference on Computer Systems and Applications, January 2005.

[9] IETF. *Home Automation Routing Requirements in Low Power and Lossy Networks (2010)*. http://tools.ietf.org/html/draft-ietf-roll-home-routing-reqs-11/. [Online; accessed 13 July 2012].

[10] Zolertia Z1. *Wireless Sensor Node*. http://zolertia.sourceforge.net/wiki/images/e/e8/Z1_RevC_Datasheet.pdf. [Online; accessed 16 January 2014].

6

Broadcast-Based QoS-Aware In-Network Aggregation for Local Wireless Sensor Network Traffic

6.1 Introduction

In this chapter, we investigate how in-network aggregation can be performed for local sensor network traffic. This is traffic between nodes in a single sensor network. As can be seen in Fig. 6.1, local sensor network traffic is dominated by point-to-point and multipoint-to-multipoint communication. As a consequence, nodes are often organized by a peer-to-peer or (full) mesh topology.

In Chapter 5, we have shown that the unicast-based QoS-aware tunable protocol-independent in-network aggregation scheme performs well when there is one sink node, but performance decreases when different sink nodes are addressed. As a consequence, this approach is not suited when traffic has to be routed inside a sensor network where many different sensor nodes can communicate with many different other sensor nodes. This traffic pattern leads to fewer aggregation opportunities since the queues on the nodes become filled with packets with many different destinations. As a consequence, more energy is wasted and the QoS level is decreased since packets will have to wait longer resulting in a higher drop probability.

Therefore, in this Chapter, we propose to use broadcast aggregation to aggregate packets independent of their next-hop destination. We show that applying

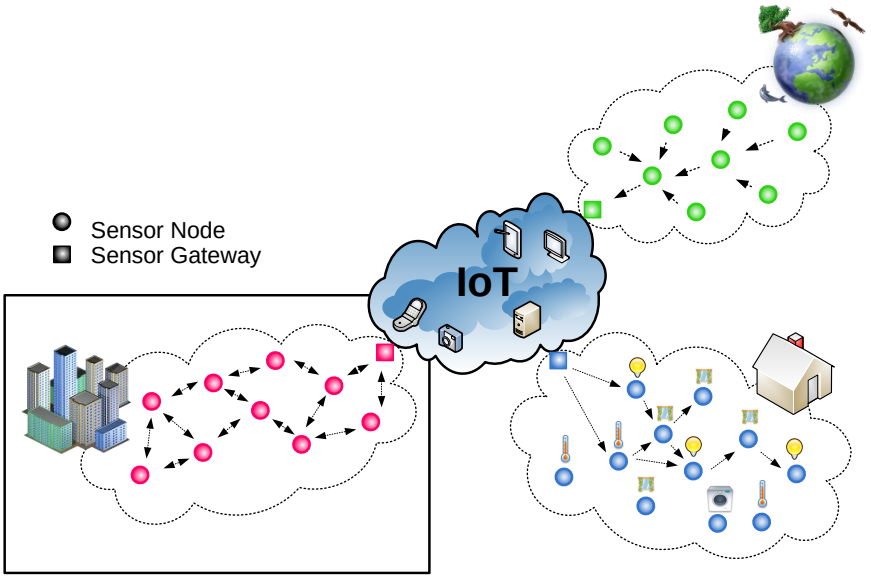


Figure 6.1. Local WSN traffic: traffic flows between nodes inside a sensor network

broadcast aggregation leads to faster aggregation with a higher overall QoS level and a lower energy consumption.

The remainder of this chapter is organized as follows. Section 6.2 gives an overview of the related work on QoS-aware broadcast in-network aggregation, while in Section 6.3 the applied broadcast mechanism is explained. A definition and problem statement is given first, followed by a performance analysis. Simulation results are discussed in Section 6.4. We end this chapter with a conclusion in section 6.5.

6.2 Related Work

In this section, we give an overview of current QoS-aware in-network aggregation solutions that consider broadcast-aggregation.

Broadcast-based in-network aggregation takes advantage of the density of sensor networks in which one sensor node often has many neighbors. By broadcasting a packet, the packet can be received by many neighboring nodes, and for this reason, this approach is often used to increase reliability for multi-path routing.

Most broadcast-based in-network aggregation techniques in literature focus however on lossy aggregation (by using aggregation functions such as min, max and average) and sending this aggregated value by multiple paths to the sink. A focus hereby is developing duplicate-sensitive aggregation functions since broadcast

aggregation can lead to incorrect values at the sink node [1–4].

In AIDA [5], a lossless adaptive application-independent data aggregation mechanism is presented. This solution contains an aggregation module that resides between the data link and network layer. Aggregation decisions are made in accordance with an adaptive feedback-based packet-scheduling scheme that dynamically controls the degree of aggregation in accordance with the MAC delay. This dynamic feedback scheme is based on the overall queuing delay imposed on AIDA payloads that are waiting for transmission. The default degree of aggregation is 1, while if the traffic builds up, a greater degree of aggregation is allowed prior to sending. If the packets that are ready to be aggregated are targeting the same next-hop node, AIDA sends a manycast packet (or a unicast packet in the case that there is only one packet) with the target node specified. However, when there are network packets that have to be aggregated with different next-hop addresses, these packets are aggregated into a single packet and the MAC broadcast address is used as destination. A drawback of AIDA is that it only tries to reduce end-to-end delay, and energy consumption is more considered as benefit then as trade-off value. Our focus is on energy reduction within the QoS constraints.

6.3 Broadcast-Based Aggregation

6.3.1 Definition and Problem Statement

In traditional unicast aggregation, many small packets with the same (next-hop) destination are aggregated into one big packet and sent by unicast to the next-hop node along the routing path. In typical source-to-sink applications such as temperature monitoring, there are many aggregation possibilities since many packets are routed to the sink. A high degree of aggregation is possible and, as a consequence, much energy can be saved and QoS is only marginally affected.

However, in the IoT, a huge amount of devices may be interconnected with many bindings between individual devices (e.g. sensor-actuator interactions) and as a consequence, many different nodes can send packets to many different destination. Each intermediate routing node may contain many packets that have to be routed to different destinations. This leads to fewer aggregation candidates, so packets will have to be routed without being aggregated, or with a lower degree of aggregation. However, when both the timeout time and the predefined DoA_{max} value (= the maximum number of packets that can be aggregated into one single packet) are not yet reached, packets will be dropped because the queue is fully occupied with packets that are waiting for the required number of aggregate candidates or for their timeout time. This is illustrated in Fig. 6.2. Alternatively, instead of dropping packets, we could also send packets before their timeout time, but this will again increase the energy consumption. Furthermore, more packets that are

not aggregated will lead to more packets in the air and a higher drop probability due to channel issues (collisions etc.).

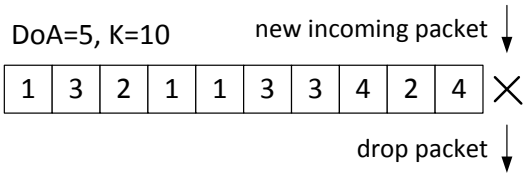


Figure 6.2. Queue overflow with predefined $DoA_{max} = 5$ and maximum queue size $K = 10$. A packet drop occurs since a new packet enters the system while the queue is fully occupied with packets that are waiting for the required number of aggregate candidates or for their timeout time.

To overcome these issues, we propose to use broadcast aggregation. Instead of aggregating packets with the same (next-hop) destination and sending them by unicast, packets are aggregated independent of their (next-hop) destination and sent by broadcast. This decouples aggregation from the routing path. In unicast communication, packets are sent to a single destination on the routing path, while by broadcast communication, a transmitted packet is received by every node within the coverage area. The receiving nodes can then extract the packet and retrieve the packet parts that are destined for them. This broadcast aggregation mechanism is visualized in Fig. 6.3.

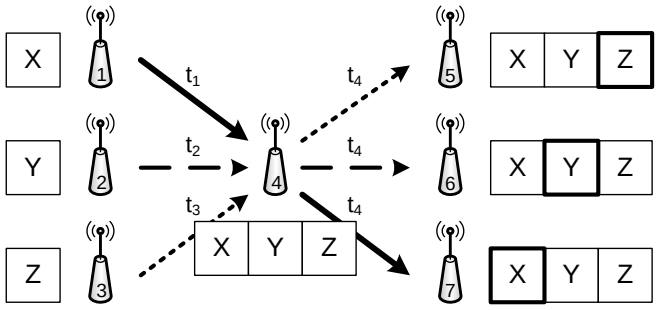


Figure 6.3. Broadcast Aggregation Mechanism: nodes 1, 2 and 3 send data packets (respectively X, Y and Z) sequentially ($t_1 < t_2 < t_3$) towards respectively nodes 7, 6 and 5. These packets wait in the intermediate node (node 4) until the requested DoA is reached and then, they are aggregated independent of their next-hop destination. This aggregated packet is then sent by broadcast on t_4 to the neighboring nodes. Destination nodes 5, 6 and 7 can then extract the data part that is destined for them (respectively Z, Y and X).

6.3.2 Performance Analysis

In the following, we compare broadcast aggregation with unicast aggregation. As already explained in Section 6.3.1, broadcast aggregation is performed as soon as the number of available packets reaches the predefined DoA_{max} value or when the maximum timeout time has reached.

To summarize, broadcast aggregation is performed:

- When there are DoA_{max} packets in the system, DoA_{max} packets are aggregated and sent by broadcast $\Rightarrow B = DoA_{max}$
- When fewer than DoA_{max} packets are in the system and the maximum timeout time has reached for at least one of the packets $\Rightarrow B < DoA_{max}$

With B the number of packets that are aggregated in a single packet that is broadcasted to all 1-hop neighbors.

For unicast aggregation, it is not possible to select the first DoA_{max} packets since these packets can have different next-hop destinations.

Unicast aggregation is therefore performed:

- When there are DoA_{max} packets *with the same next-hop destination* in the system $\Rightarrow U = DoA_{max}$
- When fewer than DoA_{max} packets *with the same next-hop destination* are in the system and the maximum timeout time has reached for at least one of the packets *with the same next-hop destination* $\Rightarrow U < DoA_{max}$

With U the number of packets that are aggregated in a single packet that is sent to a single next-hop node using unicast.

To compare the broadcast aggregation with unicast aggregation, we can use the $M/M/1/GD/K/\infty$ queuing system. This is a queuing system in which the interarrival times and service time are exponentially distributed with rate λ and rate μ respectively. There is only one serving unit, there is a general queue discipline (GD) and the source population is assumed to be infinite. However, only K packets can enter the system. This queuing system is shown in Fig. 6.4.

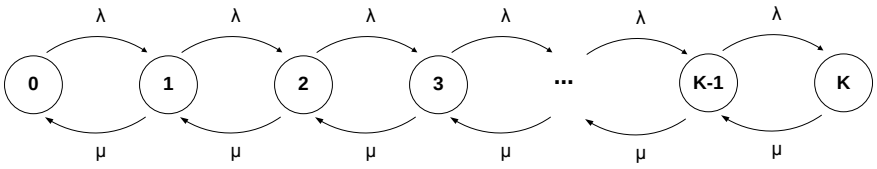


Figure 6.4. $M/M/1/GD/K/\infty$ queuing system

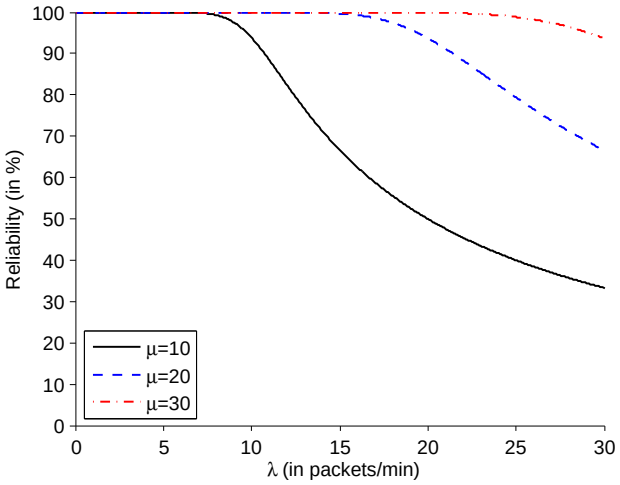


Figure 6.5. Reliability analysis for different μ values (with μ the service rate, λ the interarrival rate and $K = 15$ as the maximum queue occupation).

For $\lambda \neq \mu$ and with $\rho = \frac{\lambda}{\mu}$ the steady-state probabilities are given by:

$$\begin{aligned} p_0 &= \frac{1-\rho}{1-\rho^{K+1}} \\ p_i &= \rho^i p_0 \qquad\qquad (i = 1, 2, ..., K) \\ p_i &= 0 \qquad\qquad (i = K+1, K+2, ...) \end{aligned} \tag{6.1}$$

6.3.2.1 Reliability

In a $M/M/1/GD/K/\infty$ queuing system, the drop probability equals the probability that a new packet arrives when there are already K packets in the queue. Substituting p_0 in p_K, the reliability can be expressed as:

$$R = 1 - p_K = 1 - \frac{\rho^K(1-\rho)}{1-\rho^{K+1}} \tag{6.2}$$

In Fig. 6.5, the reliability (or the chance that a packet is not dropped from the queue) is shown for different μ values. The evaluation of Eq. (6.2) is done for $K = 15$.We can see from the figure that doubling the service rate can significantly improve the reliability, or in other words, doubling the service rate leads to an increased load capacity before a packet will be dropped.

So when broadcast aggregation is applied and the maximum timeout time has not passed, packets can be aggregated as soon as there are enough packets to meet

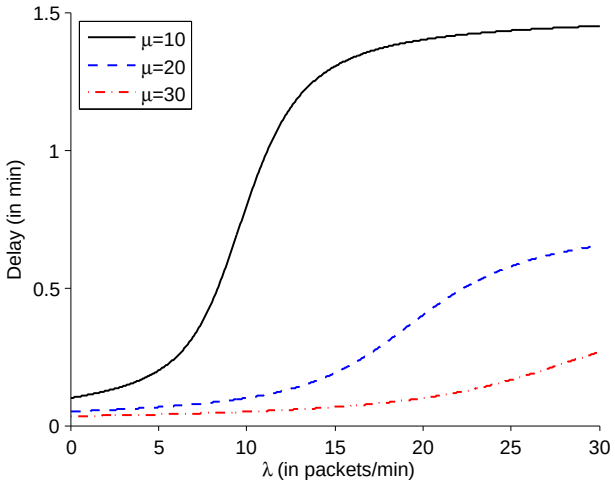


Figure 6.6. Delay analysis for different μ values (with μ the service rate, λ the interarrival rate and $K = 15$ as the maximum queue occupation).

the DoA_{max} value which will prevent from queue overflows. In unicast aggregation, the network topology determines the service rate since when there are many different streams with different next-hop destinations, it can take some time before there are enough packets with the same next-hop destination before aggregation can occur. So while B and U are the same, B will be reached earlier, which releases the queue earlier and increases the reliability.

In this section, we mentioned the node reliability caused by queue overflows. The end-to-end reliability will also be influenced by specific protocol implementations and channel related issues (e.g. collisions).

6.3.2.2 Delay

The delay or the average time that a packet resides in a node can be calculated from the Little's queuing formula:

$$D_{avg} = \frac{\overline{N}}{\lambda_{eff}} \tag{6.3}$$

with $\overline{N}$ the average number of packets in the node and λ_{eff} the average number of packets that actually enter the node. Recall that some packets may arrive in a fully occupied queue and will hence be dropped. These packets are not considered in λ_{eff}.

λ_{eff} can be calculated as:

$$\lambda_{eff} = \sum_{k=0}^{K} \lambda_k p_k = \lambda(1 - p_K) \tag{6.4}$$

and $\overline{N}$ can be calculated as:

$$\overline{N} = \sum_{k=0}^{K} k \cdot p_k \tag{6.5}$$

Combining Eq. (6.3) with Eqs. (6.4), (6.5) and (6.1), the average delay D_{avg} that a packets resides inside the system can be expressed as follows:

$$D_{avg} = \frac{\rho\left[1 - (K+1)\rho^K + K\rho^{K+1}\right]}{(1-\rho^{K+1})(1-\rho)\lambda(1 - \frac{\rho^K(1-\rho)}{1-\rho^{K+1}})} \tag{6.6}$$

In Fig. 6.6, the average delay is given for different μ values, that are expressed in terms of the initial arrival rate λ. The evaluation of Eq. (6.6) is done for $K = 15$. We can see from the figure that doubling the service rate can significantly reduce the average delay that a packet resides inside the system. So again, while B and U are the same, B will be reached earlier, which decreases the average (end-to-end) delay.

6.3.2.3 Throughput

The end-to-end throughput of the network is expressed as the average rate of successfully delivered data from source to destination. This end-to-end throughput is influenced by two parts: a node throughput and a channel throughput. The node throughput equals the service rate μ. The channel throughput depends on channel overhead, transmission errors, etc. Aggregation in general is beneficial for channel utilization since fewer packets have to contend for the medium which leads to fewer packet drops.

6.3.2.4 Energy

In Chapter 4, we have shown that energy consumption can be reduced when aggregation is applied together with a sleep-wake-up scheme. Aggregation leads to fewer transmissions and receptions which results in more sleep opportunities and a reduced energy consumption level. The transmission energy reduction was calculated as:

Parameter	Value
Simulation time	2700 s
Simulation field	75 m x 75 m
Number of nodes	256
Node deployment	16x16 grid
Routing protocol	DYMO
MAC protocol	tunable CSMA based MAC (Castalia) with duty cycle 0.2 (100 ms listening/400 ms sleeping)
Radio	CC2420 with no transmission errors
Wireless channel	no interference
Collision domain	$\pm$ 16 m, depends on the receiver signal strength ($\pm$ 2 hops)
Queue size (K)	15
DoA	5

Table 6.1. Local WSN traffic: simulation settings

$$E_{tx}^{reduction} \approx$$

$$1 - \frac{E_{TO} + E_H^A + DoA_{avg} E_{DATA}^{NA}}{E_{TO} + E_H^{NA} + E_{DATA}^{NA}} \frac{1}{DoA_{avg}} \quad (6.7)$$

in which E_{TO} is the energy consumption caused by transmission overhead per packet transmission. This contains the CCA (Clear Channel Assessment) time, the RX-TX turnaround time but also the overhead created by the MAC protocol, e.g. by beacons. E_H^{NA} is the energy consumed to send a not aggregated packet header, E_H^A is the energy consumed to send a header of an aggregated packet and E_{DATA}^{NA} is the energy consumed to send individual data parts.

From Eq. (6.7), we can see that the energy reduction depends on the average DoA level (DoA_{avg}). In general, the more packets that are aggregated (= a higher DOA_{avg}), the more energy that will be saved. Because in broadcast aggregation more packets are faster aggregated, more energy will be saved.

However, the total energy consumption will also depend on the applied sleep-wake-up scheme and the used MAC protocol. Since the impact is very protocol specific, this will be further discussed in the simulation results.

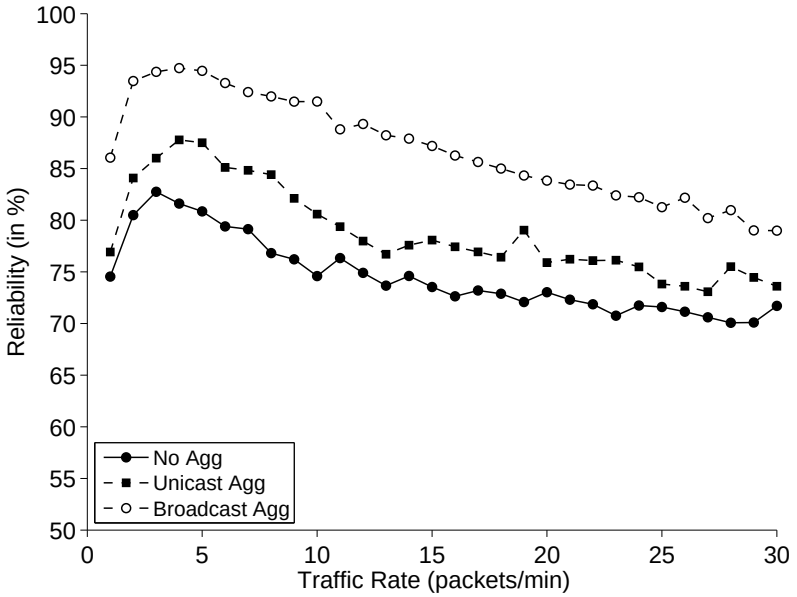


Figure 6.7. End-to-end reliability

6.4 Simulation Results

Simulations are again performed using CastaliaWe use a network scenario with 256 nodes, deployed in a square grid with size 75 m. This setup was again chosen in order to meet the minimal requirements of RFC 5826 for home automation routing in low power and lossy networks. The DYMO protocol is chosen as routing protocol and a tunable MAC protocol with sleeping scheme is chosen as MAC protocol. Simulations are performed with $DoA = 5$ and with maximum queue size $K = 15$.

In our scenario, we consider a multipoint-to-multipoint application in which each node (except the destination nodes) is transmitting packets to one of 20 randomly chosen destination nodes. A simulation lasts again 3600 seconds and statistics are again generated in steady state between 300 and 3000 seconds for different average traffic rates (from 1 to 30 packets/min) as given on the x-axis of the figures. The maximum packet timeout was again set to 10 seconds. The used simulation settings can be found in Table 6.1.

6.4.1 Reliability

Fig. 6.7 shows the overall reliability as the number of application packets received compared to the number of application packets sent. Figs. 6.8 and 6.9 show re-

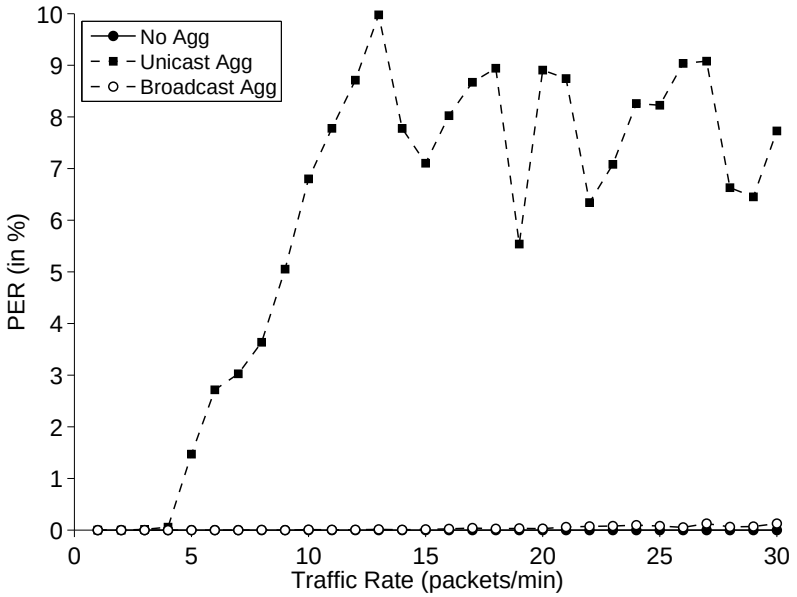


Figure 6.8. Packet Error Rate (PER): Number of dropped packets in the queue compared with the number of packets sent

spectively the number of packets dropped in the queue and the number of packets dropped due to the lower network layers (Radio, MAC, Physical channel). The packet error rate (PER) is expressed as the number of application packets dropped compared to the number of application packets sent. Packets in the queue are dropped due to queue overflows, while packets dropped by the lower layers are caused by the fact that a sensor node cannot simultaneously send and receive, packets can collide etc.

From Fig. 6.8, we can see that for unicast aggregation, starting from an average traffic rate of 3 packets/min, packet drops occur in the queue. This is not the case in the no aggregation and broadcast aggregation scenario, since in the no aggregation scenario, packets don't have to wait for aggregation candidates and can be sent immediately. In the broadcast aggregation scenario, aggregated packets can be sent as soon as there are 5 packets in the queue. This effect can be seen in Fig. 6.10 that shows the average queue occupation. We can see the average queue occupation is on average 2 packets lower for broadcast aggregation than for unicast aggregation.

Fig. 6.9 shows on its turn the impact of the lower network layers on the reliability. The figure shows that in the beginning the packet error rate (PER) is relative high, then drops, and finally slowly increases again. The high PER at low traffic rates can be explained by the fact that each node will have approximately the same packet rate. Remember that the packet rate on the x-axis is the average

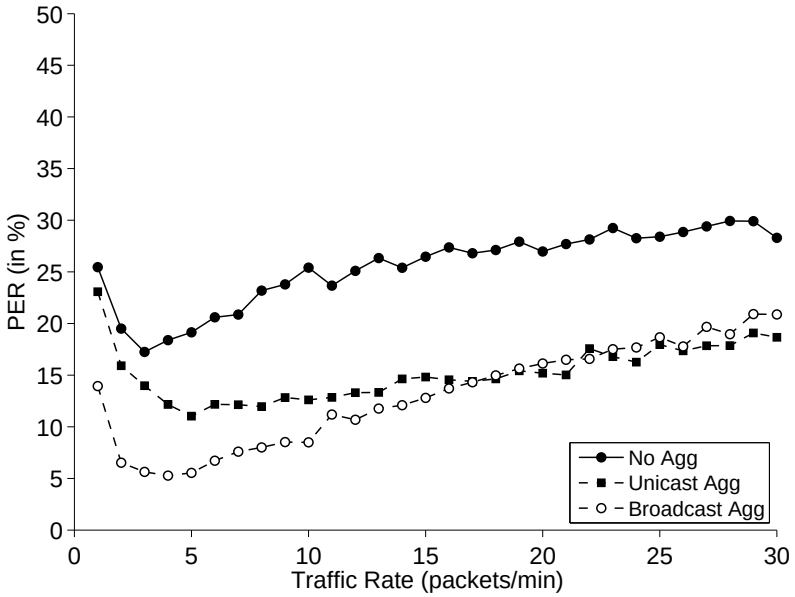


Figure 6.9. Packet Error Rate (PER): Number of dropped packets due to the MAC protocol compared with the number of packets sent

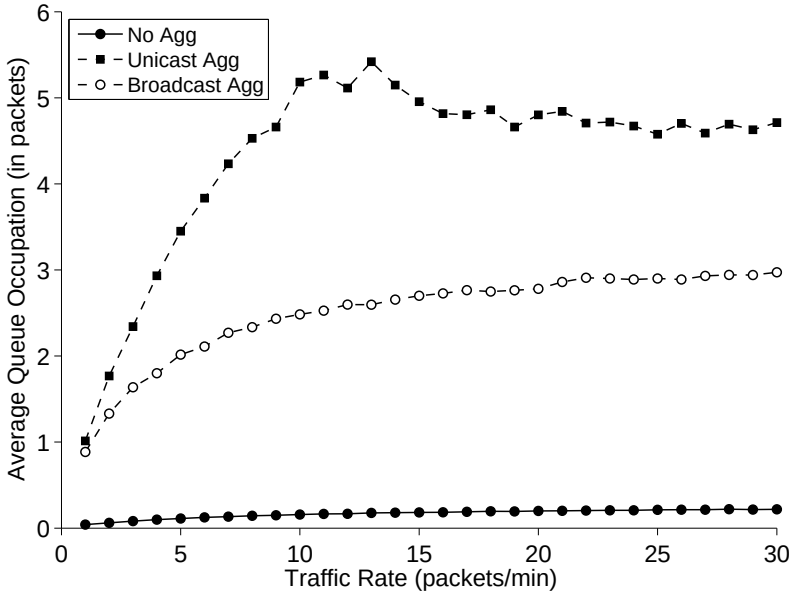


Figure 6.10. Average Queue Occupation

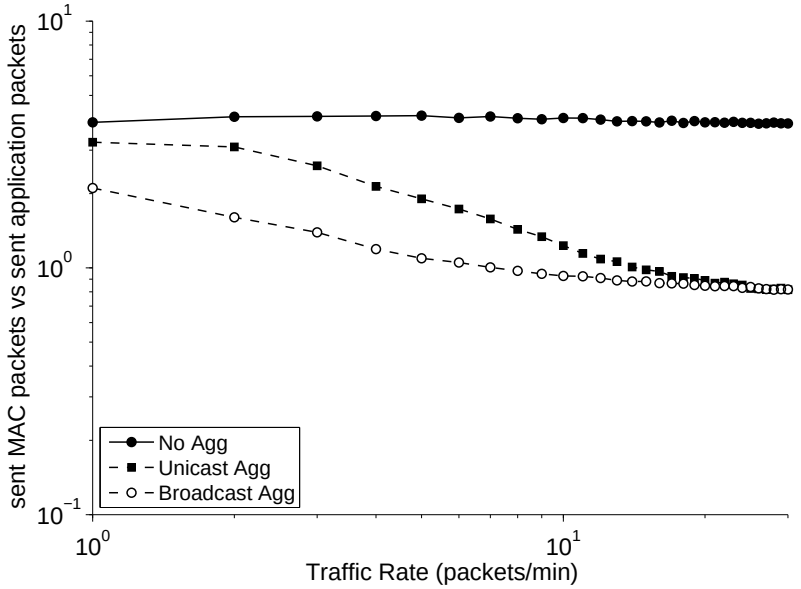


Figure 6.11. Ratio of MAC data packets sent to application packets sent

packet rate of all the nodes. So at low packet rates, most sensor nodes will have the same packet rate, and will try to enter the medium on the same time. As a consequence, more packets will be lost. From an average traffic rate of 5 packets/min, the PER starts increasing since more packets are in the air, which leads to a higher PER. Furthermore, we can see that the PER is higher in the no aggregation scenario, since when packets are not aggregated, more packets are in the air and more packets can be lost.

Since both unicast and broadcast aggregation combine packets into one big packet, we should expect that the packet loss due to the lower network layers should be approximately the same. This is however not the case at low traffic rates. This can be explained by a fact to which we refer in the following as 'partial aggregation'.

A 'partly aggregated' packet is a packet that contains fewer than DoA aggregated packet parts. For instance, when the DoA was set to 5, a partly aggregated packet will contain fewer than 5 aggregated packet parts. This effect is mainly caused by the timeout time that was introduced. At low traffic rates, the timeout time will pass before DoA packet parts can be aggregated. At this moment, the aggregated packet will be sent with as much packet parts that are available. This partial aggregation has a cascading effect. When a packet with DoA packet parts is sent, on the following node, there are obviously enough packets parts to send a new aggregated packet to the next-hop node. However, when a partly aggregated

packet was received, this node will on its turn wait again on DoA packet parts, and again, the timeout time can pass. Partly aggregated packets however mean more packets in the air and an increased chance on packet drops. The higher the load in the network, the fewer partly aggregated packets that exist, and the fewer packet drops that occur. We indeed observe in Fig. 6.9 that for higher loads the PER is similar for unicast and broadcast aggregation. Fig. 6.11 shows the ratio of MAC data packets sent to application packets sent. This ratio can be higher than 1 since the MAC data packets are measured through the network, and thus a MAC packet is measured on each intermediate node, while the application packets are only measured on the sending node. The figure clearly shows the effect of partial aggregation at low traffic rates. For the no aggregation scenario, the ratio equals the average number of hops, which is 4. When packets are aggregated, this ratio decreases and for high traffic rates, we can see that this ratio drops below 1 because there, maximal aggregation (up to 5 packets) occurs. We can see that with broadcast aggregation, partial aggregation is reduced since more packets will be faster aggregated.

Looking back at Fig. 6.7, we can see that the end-to-end reliability of broadcast aggregation is increased up to 23% compared with the no aggregation scenario and up to 15% compared with the unicast aggregation scenario. Although there are no packet drops in the queue in the no aggregation scenario, we see that the reliability is significant lower than the broadcast aggregation scenario. This has been explained earlier by the fact that with broadcast aggregation, fewer transmissions occur which is beneficial for the channel occupation and as a consequence, the reliability increases.

6.4.2 Delay

Fig. 6.12 displays the average end-to-end packet delay. This packet delay is measured on the individual application packets, not on the aggregated packets. We can see that with broadcast aggregation, the delay can be reduced up to 52% compared with unicast aggregation. Delay is of course higher than with no aggregation since with aggregation, packets are waiting in the queue for a certain period in order to have fewer transmissions and save energy.

6.4.3 Throughput

In Fig. 6.13, the end-to-end throughput of the network is demonstrated. This throughput is expressed as the average number of successful received packets per second per node. We can see at an average traffic rate of 10 packets/min, up to 1.88 packets/min are more received with broadcast aggregation compared with no aggregation and at a traffic rate of 12 packets/min, up to 1.59 packets/min are more

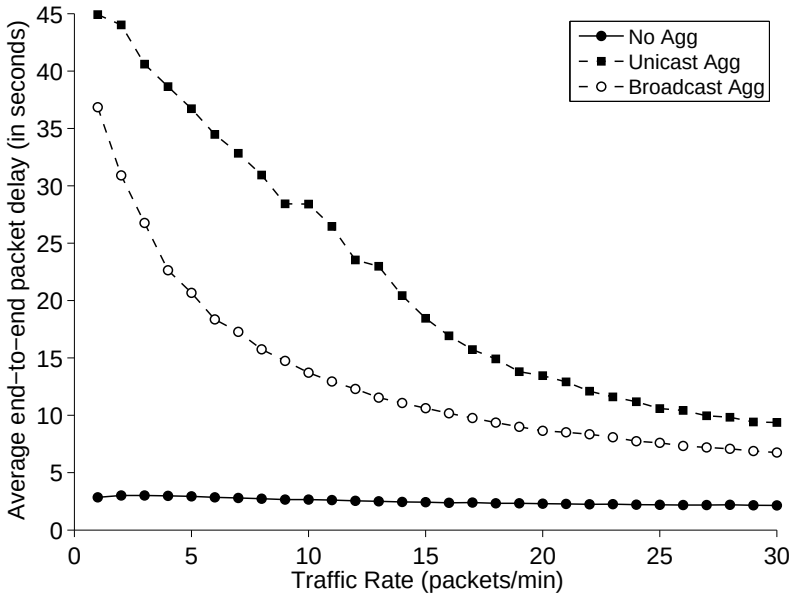


Figure 6.12. Average end-to-end packet delay

received with broadcast aggregation compared with unicast aggregation. This is mainly caused by faster transmission and less packet loss.

6.4.4 Energy

Fig. 6.14 shows that the total transmit energy consumption with broadcast aggregation can be reduced up to 29% compared to unicast aggregation and up to 71% compared to no aggregation. Fig. 6.17 shows the actual DoA at which aggregation is performed. This actual degree of aggregation can be lower than the theoretical maximum aggregation rate due to the partial aggregation as explained in Section 6.4.1. Fig. 6.17 shows that with broadcast aggregation, the average degree of aggregation is on average 1 packet higher.

From Eq. (6.7) and the values in Table 3.1, we could calculate that theoretically up to 74% of energy could be saved when broadcast aggregation with $DoA_{avg} = 5$ is applied compared with no aggregation. The difference between the theoretical and the actual energy reduction could be explained by the fact that in our simulation $DoA_{avg} = 5$ is not reached. Furthermore, we can see that starting from an average traffic rate of 21 packets/min, broadcast aggregation consumes more energy than unicast aggregation. This is because lost packets are not taken into account. Indeed, with unicast aggregation, more packets are lost and since lost packets are not further routed, less energy is consumed than expected.

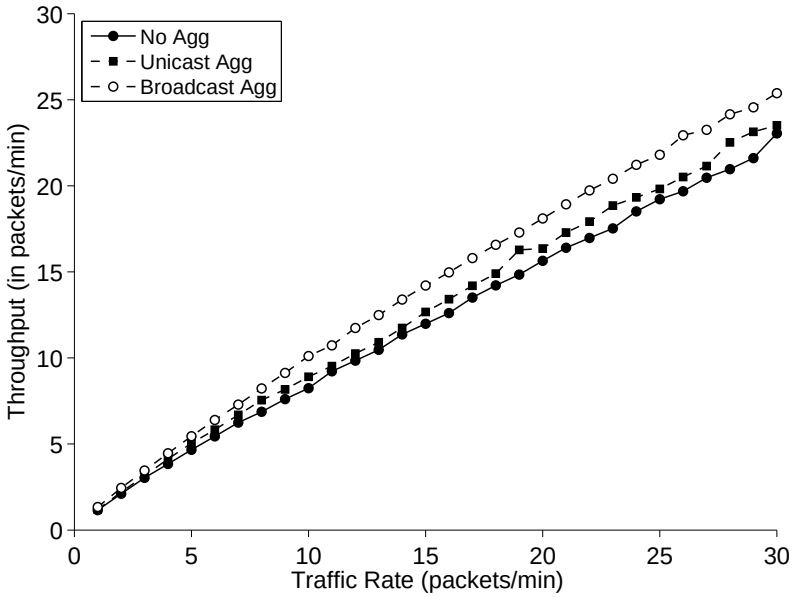


Figure 6.13. End-to-end data throughput per node

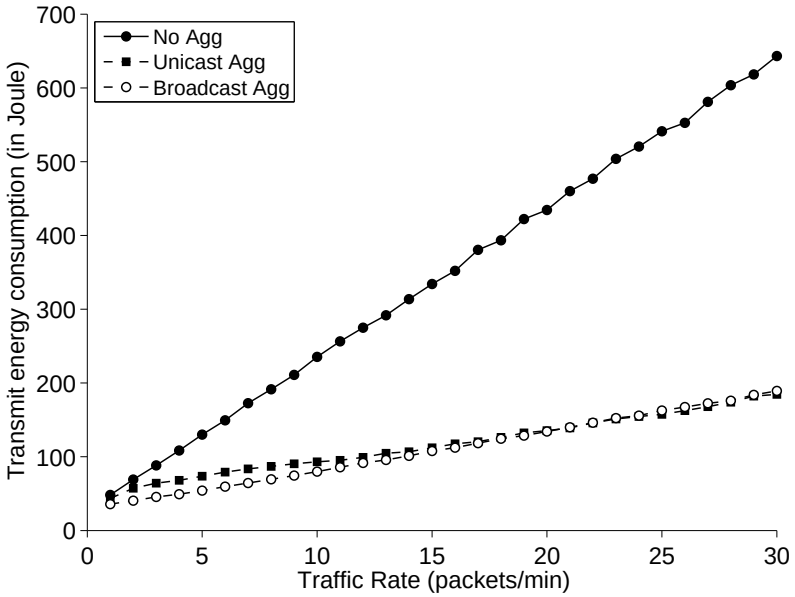


Figure 6.14. Total transmit energy consumption

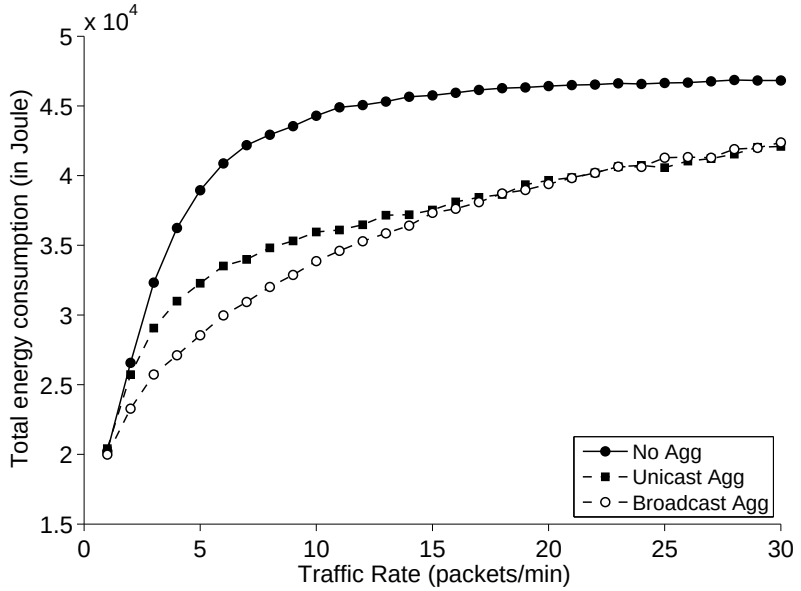


Figure 6.15. Total energy consumption

Figs. 6.15 and 6.16 show respectively the total energy consumption and the average energy consumption per successfully delivered packet.

Taking into account the total energy consumption, 27% of energy can be saved compared with no aggregation and up to 13% compared with unicast aggregation.

When we take a look at the energy per successfully delivered packet (Fig. 6.16), we can see that up to 38% of energy can be saved compared with no aggregation and up to 19% compared with unicast aggregation.

From Figs. 6.14 and 6.15, we see that the total energy reduction gain is much lower than the total transmit energy reduction gain. The reason can be found in the used MAC protocol and can be seen in Figs. 6.18 to 6.20 that shows the energy distribution of no aggregation, unicast aggregation and broadcast aggregation in terms of transmit energy, idle listening energy, receive energy and sleep energy. We can see that much energy is lost due to idle listening.

A detailed overview of the energy contributions can be found in Figs. 6.21 to 6.22.

The maximum idle listen time with no aggregation in Fig. 6.21 is due to the used MAC protocol and the load. First, the number of sent packets increases, so the number of sent beacons also increases and nodes are longer awake. As a consequence, the idle listen interval increases. However, from a certain point, the idle listening period will decrease since part of this time will be used to transmit/forward/receive the increasing amount of packets. This behavior is also expected for

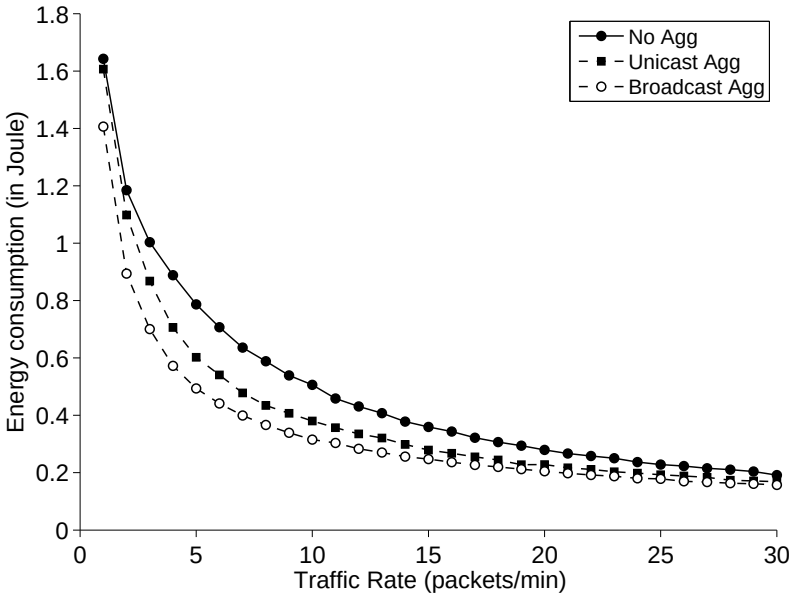


Figure 6.16. Average energy consumption per successful delivered data packet

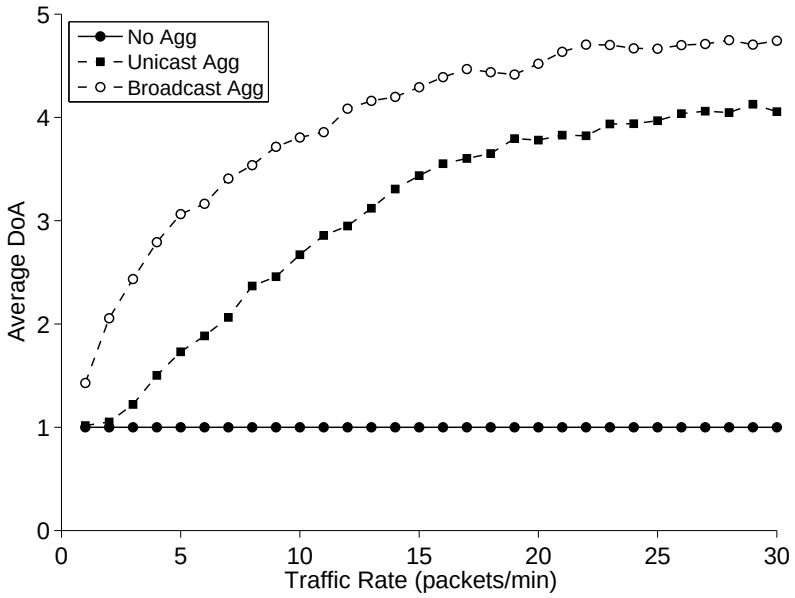


Figure 6.17. Average degree of aggregation (DoA) level at which aggregation occurs

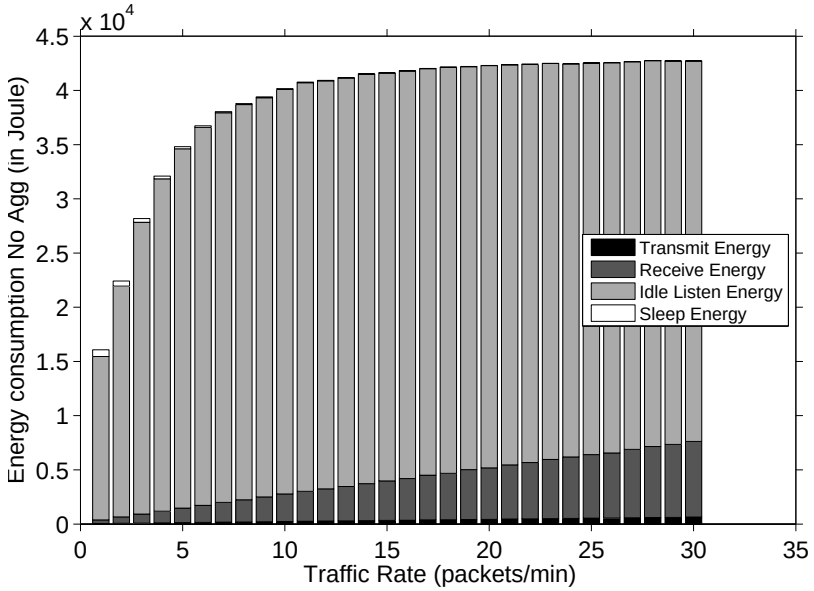


Figure 6.18. Total no aggregation energy distribution

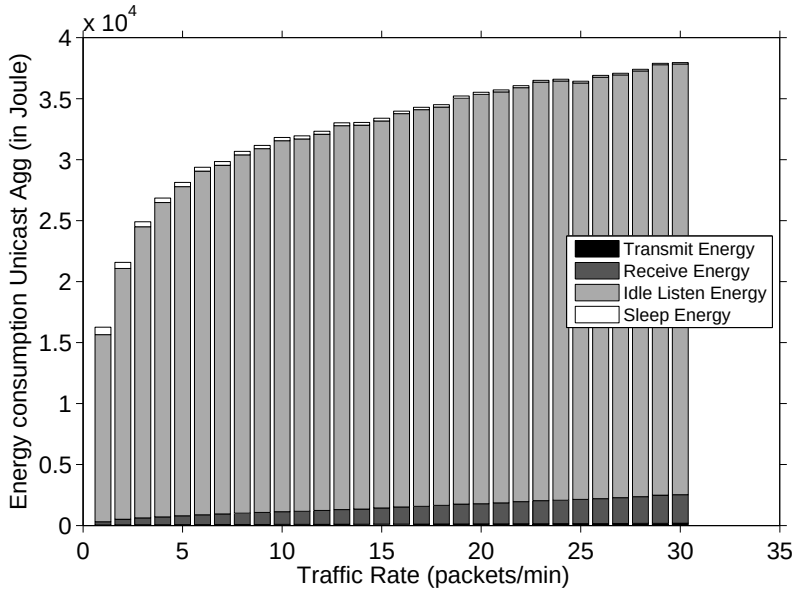


Figure 6.19. Total unicast aggregation energy distribution

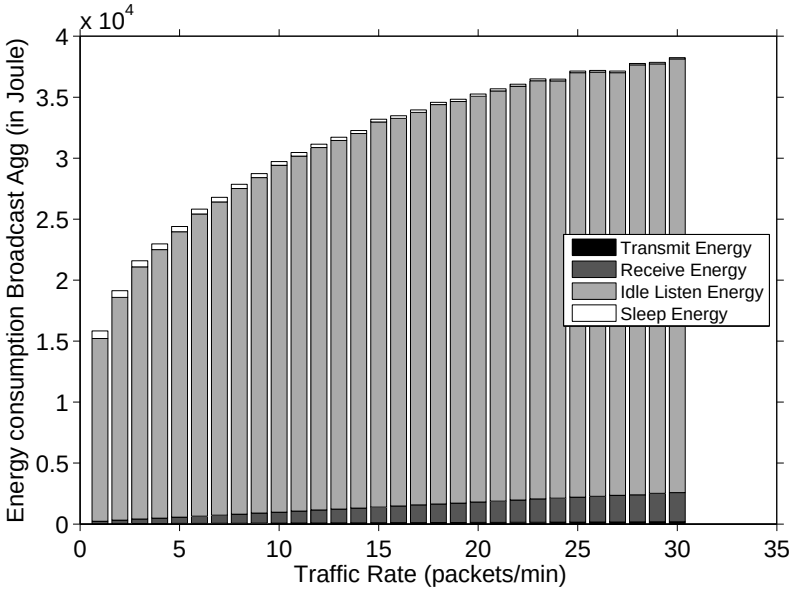


Figure 6.20. Total broadcast aggregation energy distribution

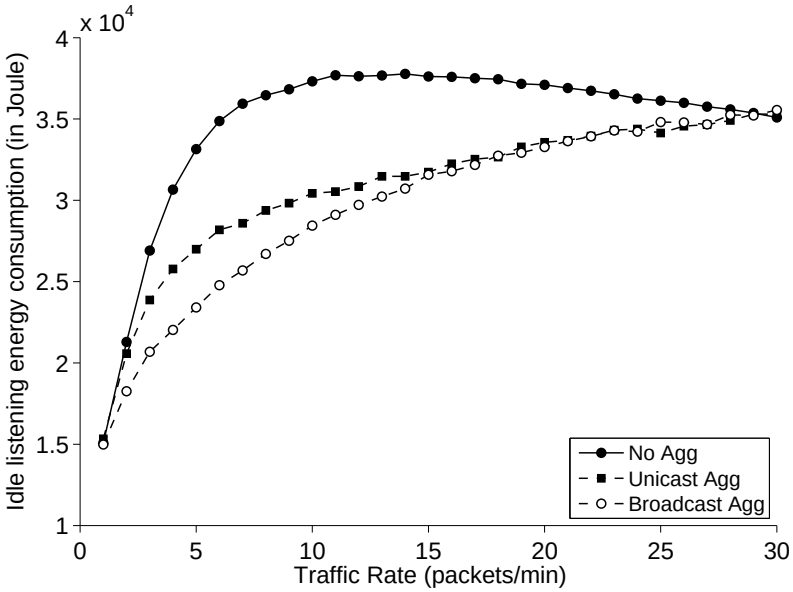


Figure 6.21. Total idle listening energy consumption

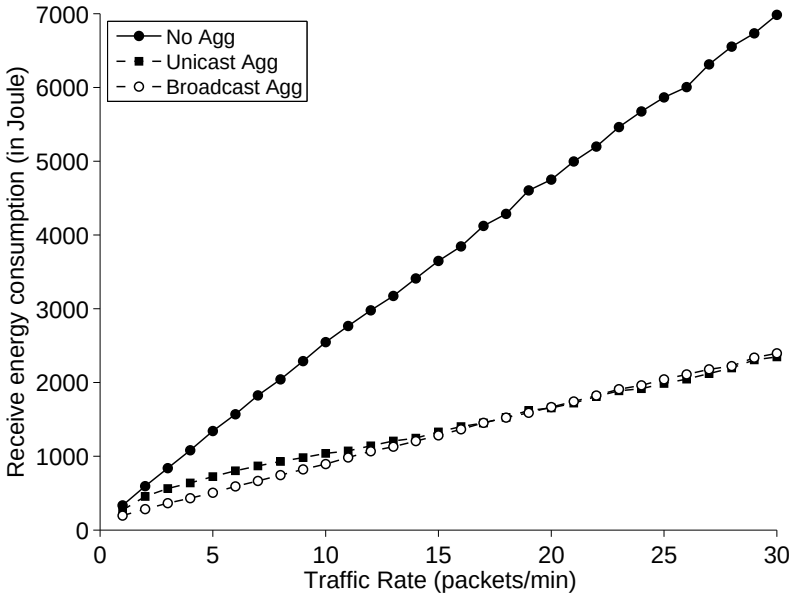


Figure 6.22. Total receive energy consumption

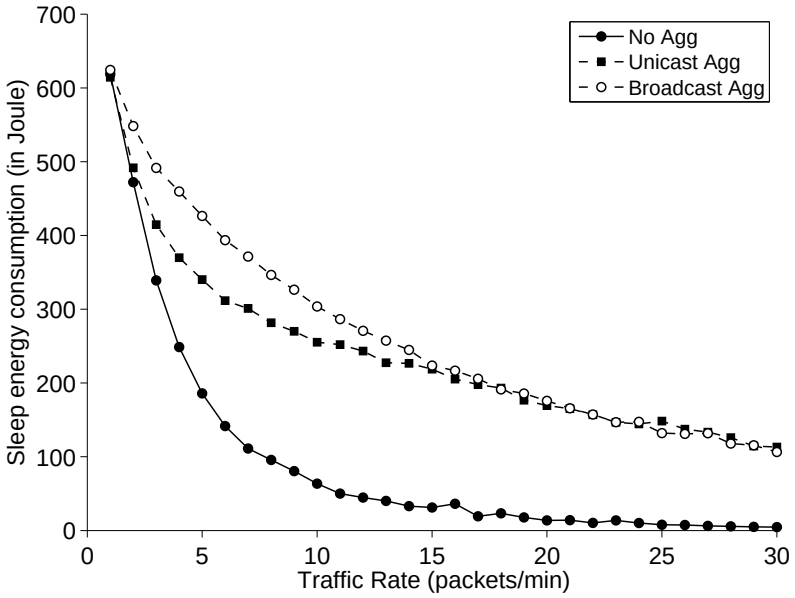


Figure 6.23. Total sleep energy consumption

unicast and broadcast aggregation, but since the amount of traffic increases slower since more packets are aggregated, this point will occur at bigger traffic rates.

Fig. 6.22 shows the total receive energy. This is the energy consumed by receiving both beacons and date, even the data that is not destined for this node.

Finally, Fig. 6.23 shows the total amount of energy that is consumed by sleeping. Both from Fig. 6.22 and Fig. 6.23, we can see that broadcast aggregation seems to perform worse than unicast aggregation starting from an average traffic rate of 21 packets/min. This is however not true since lost packets are not taken into account. Unicast aggregation leads to more lost packets which are not routed. This results in less receive energy consumption and more sleep energy consumption.

6.5 Conclusion

While the unicast-based tunable QoS-aware protocol-independent QoS-aware in-network aggregation scheme was sufficient for source-to-sink traffic with only a few sink nodes, this is not the case when sensor network traffic has to be routed between many different nodes. When there exist many connections with different destinations, aggregation becomes slower, delay increases, reliability drops and energy consumption increases.

In this chapter, we propose to use broadcast aggregation as a solution to overcome these drawbacks. We have shown that broadcast aggregation reduces the average queue occupation with 2 (of the 15 available) places, which leads to fewer packet drops. This leads on its turn to a throughput and reliability increase up to 23% compared with no aggregation and up to 15% compared with unicast aggregation. Moreover, we have shown packets become less dependent of the individual timeouts per destination which reduces the drawbacks of partial aggregation.

Furthermore we have shown that the average queue delay is decreased by 52% compared with unicast aggregation because aggregation can be performed faster.

Finally, the average degree of aggregation is higher, which leads to fewer packets, less packet overhead and as a consequence, an energy reduction up to 27% compared with no aggregation and up to 13% compared with unicast aggregation.

References

[1] S. Nath, P. B. Gibbons, S. Seshan, and Z. R. Anderson. *Synopsis diffusion for robust aggregation in sensor networks*. In Proceedings of the 2nd international conference on Embedded networked sensor systems, pages 250–262, 2004. doi:10.1145/1031495.1031525.

[2] S. Motegi, K. Yoshihara, and H. Horiuchi. *DAG Based In-Network Aggregation for Sensor Network Monitoring*. In Proceedings of the 2005 Symposium on Applications and the Internet, pages 292–299, 2006. doi:10.1109/SAINT.2006.20.

[3] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. *TAG: a Tiny AGgregation service for ad-hoc sensor networks*. SIGOPS Operating System Review, 36(SI):131–146, December 2002. doi:10.1145/844128.844142.

[4] S. Gobriel, S. Khattab, D. Mossé, J. Brustoloni, and R. Melhem. *RideSharing: Fault Tolerant Aggregation in Sensor Networks Using Corrective Actions*. In Proceedings of the 3rd Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks, pages 595–604, September 2006.

[5] T. He, B. M. Blum, J. A. Stankovic, and T. Abdelzaher. *AIDA: Adaptive Application-Independent Data Aggregation in Wireless Sensor Networks*. ACM Transactions on Embedded Computing System, Special issue on Dynamically Adaptable Embedded Systems, 3(2):426–457, 2004.

7

Broadcast-Based QoS-Aware In-network Aggregation for Wireless Incoming Sensor Network Traffic

7.1 Introduction

In this chapter, we analyze the use of in-network broadcast aggregation for incoming sensor network traffic. This is traffic that is routed from a gateway (connected with the Internet) towards one or more sensor nodes inside a sensor network. As can be seen in Fig. 7.1, this can be point-to-point and point-to-multipoint communication. As a consequence, nodes are often organized by a point-to-point or a (reverse) tree topology.

The remainder of this chapter is structured as follows. Section 7.2 gives an overview of the related work on aggregating data packets from a gateway towards sensor nodes, while Section 7.3 explains how broadcast aggregation can be used to aggregate this particular traffic type. Simulation results are discussed in Section 7.4. We end this chapter with a conclusion in Section 7.5.

7.2 Related Work

To the best of our knowledge, this is the first work that addresses the use of (broadcast) aggregation for aggregating traffic originating from a gateway towards indi-

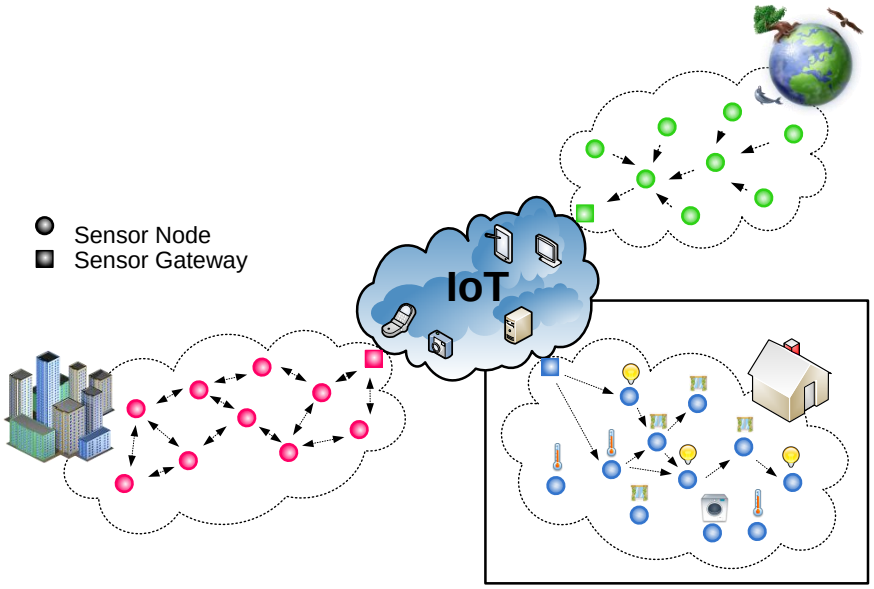


Figure 7.1. Incoming WSN traffic: traffic flows from a gateway towards the sensor nodes

vidual sensor nodes in order to reduce the overall energy consumption and increase the QoS level.

There is however an interest to have solutions for this traffic type. For instance, in [1], the authors state that in the Internet of Things, applications may need to aggregate data coming from a group of sensors or actuators in order to obtain accurate results. Depending on the application, information from individual sensors might not be sufficient, reliable or useful and/or data from several nodes should be compared or aggregated. Using multicast for this purpose has however some disadvantages. It is difficult to avoid duplication, basic multicast is not reliable and an efficient multicast implementation comes with an increased footprint. Therefore, the authors considered unicast-based group communication as an alternative to multicast-based group communication. The main downside of using unicast opposed to multicast is that more packets need to be transmitted. Our broadcast aggregation solution can be used to tackle this problem, aggregating different unicast requests into a single broadcast message.

7.3 Broadcast-Based In-Network Aggregation for Sink-to-Source Traffic

Traditionally, (unicast) in-network aggregation is used for traffic that is routed from different source nodes towards one or a few sink nodes that collect data. Since each packet has the same destination, this results in many aggregation possibilities which leads to a high energy reduction and an increased QoS level, as we have shown in Chapter 5.

In Chapter 6, we have seen that the application domain of in-network aggregation can be enlarged by applying broadcast aggregation in sensor networks where packet are routed from and to several different nodes.

This chapter focuses on traffic that is routed from a gateway towards one or more sensor nodes (also referred as sink-to-source traffic). For this traffic type, unicast aggregation can only be applied to packets with a destination that is on the same routing path. Therefore, we investigate the use of broadcast aggregation to allows aggregating packets that are on different routing paths, but that travel together during a certain amount of time before being split.

The performance of broadcast aggregation depends on the used network topology. Packets with a destination that is in each other neighborhood will have many aggregation opportunities, while packets with a destination that is far from each other will have fewer aggregation opportunities.

7.4 Simulation Results

Simulations are again performed using Castalia. We use a network scenario with 256 nodes, deployed in a square grid with size 75 m chosen in order to meet the minimal requirements of RFC 5826 for home automation routing in low power and lossy networks. The DYMO protocol is chosen as routing protocol and the tunable MAC protocol with sleeping scheme is chosen as MAC protocol.

For simulating the sink-to-source traffic, we use a point-to-multipoint application in which one gateway node transmit packets to 60 sensor nodes, as can be seen in Fig. 7.2.

Again, a simulation lasts 3600 seconds and statistics are generated in steady state between 300 and 3000 seconds for different average traffic rates (from 4 to 120 packets/min in steps of 4 packets) as given on the x-axis of the figures. The maximum packet timeout was again set to 10 seconds. The used simulation settings can be found in Table 7.1. As in Chapter 6, we investigate the difference between no aggregation, unicast aggregation and broadcast aggregation on reliability, delay, throughput and energy.

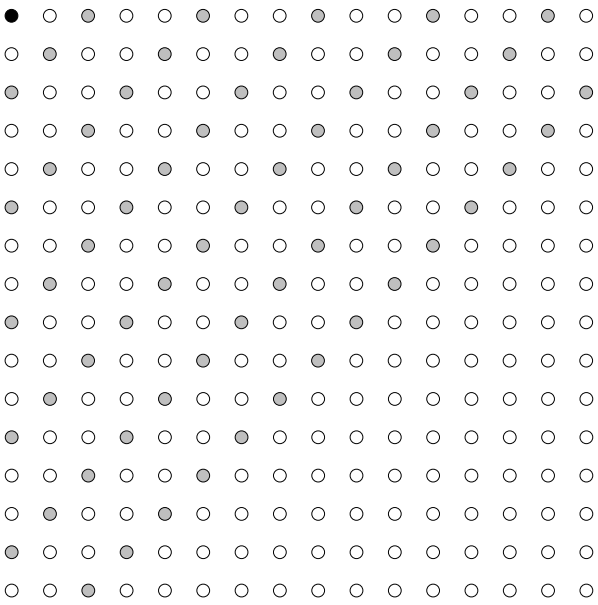


Figure 7.2. Incoming sensor network traffic topology: the black node is the gateway and the gray nodes are the receiving nodes.

7.4.1 Reliability

Fig. 7.3 shows the overall reliability as the number of application packets received compared to the number of application packets sent. Figs. 7.4 and 7.5 show respectively the number of packets dropped in the queue and the number of packets dropped due to the lower network layers (Radio, MAC, Physical channel). The packet error rate (PER) is expressed as the number of application packets dropped compared to the number of application packets sent. Packets in the queue are dropped due to queue overflows, while packets dropped by the lower layers are caused by the fact that a sensor node cannot simultaneously send and receive, packets can collide etc.

From Fig. 7.4, we can see that for unicast aggregation, starting from an average traffic rate of 28 packets/min, packet drops occur in the queue both for unicast and no aggregation scenario. In the broadcast aggregation scenario, aggregated packets can be sent as soon as there are 5 packets in the queue, while for unicast aggregation, the queue becomes filled with packets that are waiting on the required number of aggregation candidates. In the no aggregation scenario, the gateway node has to forward too many packets and the medium cannot follow. Fig. 7.6 shows the average queue occupation. We should remark that these values are however averaged over all nodes that forward packets, but the gateway and the nodes closest to the gateway will encounter higher load values.

Parameter	Value
Simulation time	2700 s
Simulation field	75 m x 75 m
Number of nodes	256
Node deployment	16x16 grid
Routing protocol	DYMO
MAC protocol	tunable CSMA based MAC (Castalia) with duty cycle 0.2 (100 ms listening/400 ms sleeping)
Radio	CC2420 with no transmission errors
Wireless channel	no interference
Collision domain	$\pm$ 16 m, depends on the receiver signal strength ($\pm$ 2 hops)
Queue size (K)	15
DoA	5

Table 7.1. Incoming WSN traffic: simulation settings

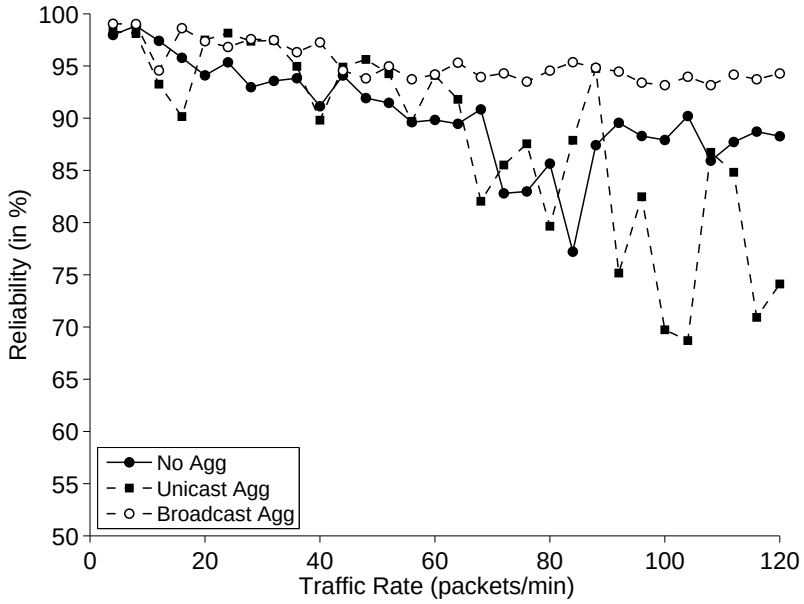


Figure 7.3. End-to-end reliability

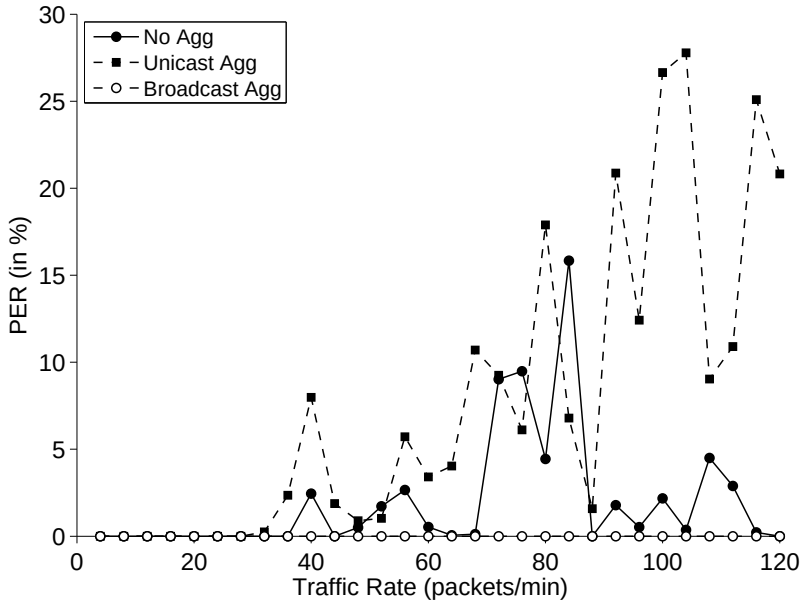


Figure 7.4. Packet Error Rate (PER): Number of dropped packets in the queue compared with the number of packets sent

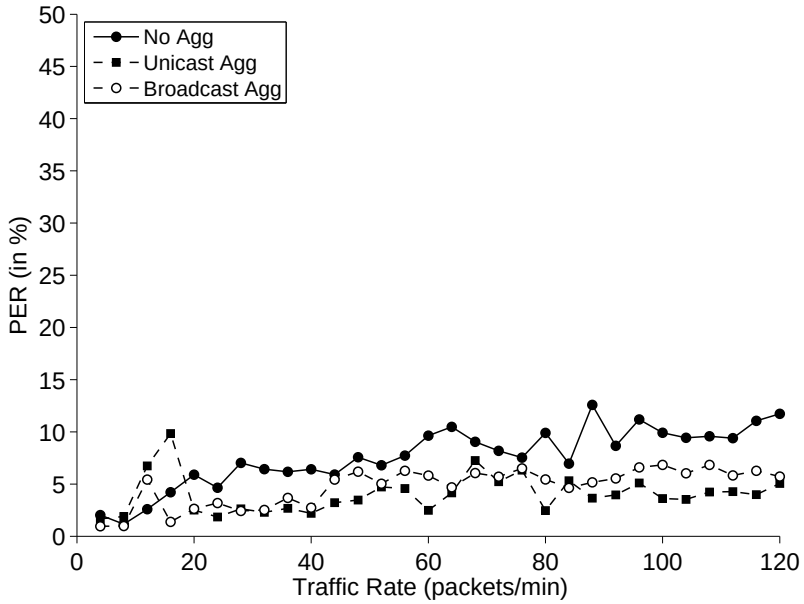


Figure 7.5. Packet Error Rate (PER): Number of dropped packets due to the MAC protocol compared with the number of packets sent

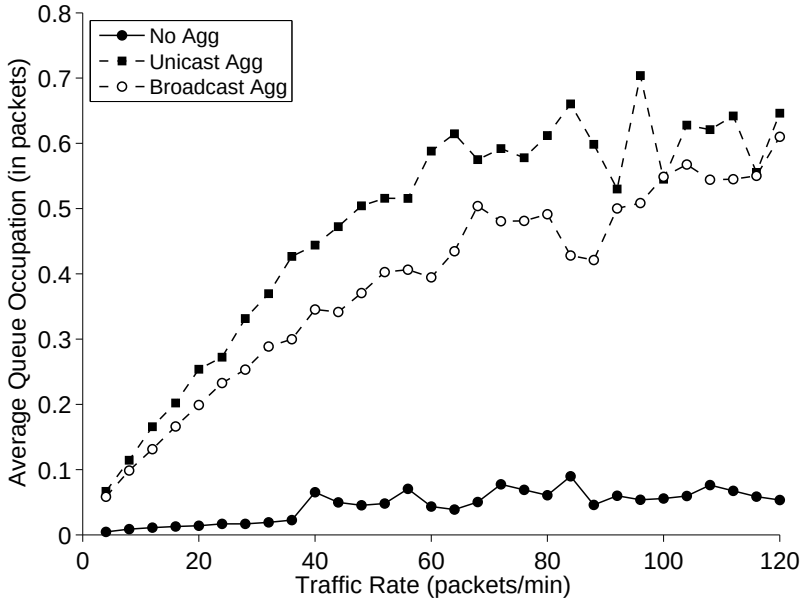


Figure 7.6. Average Queue Occupation

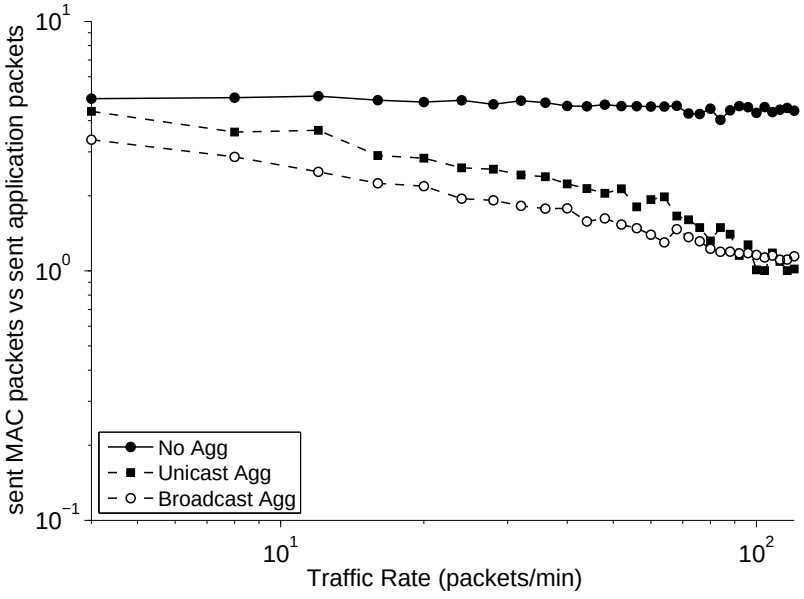


Figure 7.7. Ratio of MAC data packets sent to application packets sent

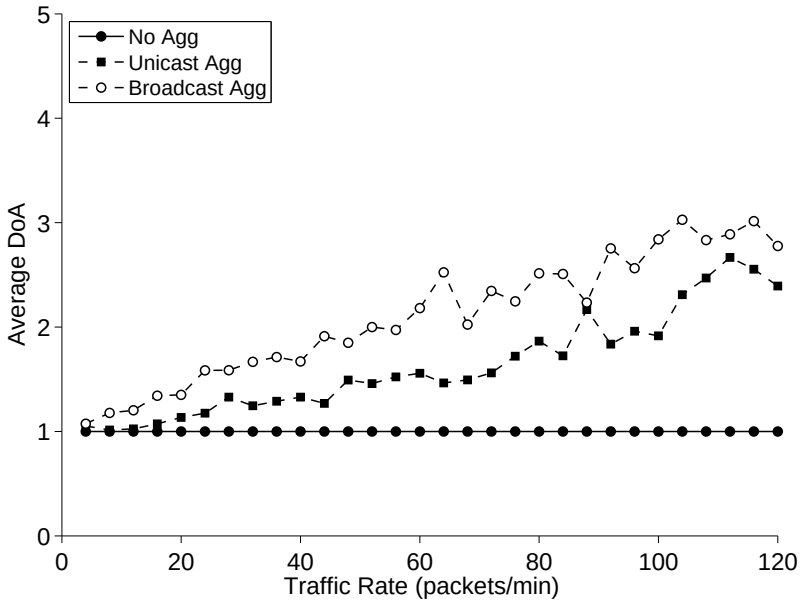


Figure 7.8. Average degree of aggregation (DoA) level at which aggregation occurs

Fig. 7.5 shows in turn the impact of the lower network layers on the reliability. The figure shows that the no aggregation scenario suffers from the highest PER. This PER is caused due to the big number of individual packets that are being sent.

Since both unicast and broadcast aggregation combine packets into one big packet, we would again expect that the packet loss due to the lower network layers would be approximately the same. This is however not the case at low traffic rates. This can again be explained by 'partial aggregation', as defined in Section 6.4.1. Furthermore, we can see that at traffic rate higher than 20 packets/min, the PER for broadcast aggregation becomes larger than for unicast aggregation. This can be clarified by the fact that in unicast aggregation, more packets are dropped, and fewer transmitted packets leads to a lower average PER since fewer packets are in the air.

From Fig. 7.7, which shows the ratio of MAC data packets sent to application packets sent, we can see that the average number of hops is 4. The higher average DoA, as can be seen in Fig. 7.8 leads to fewer sent MAC packets vs. application packets for unicast aggregation.

Looking back at Fig. 7.3, we can see that the end-to-end reliability of broadcast aggregation is increased up to 24% compared with the no aggregation scenario and up to 37% compared with the unicast aggregation scenario.

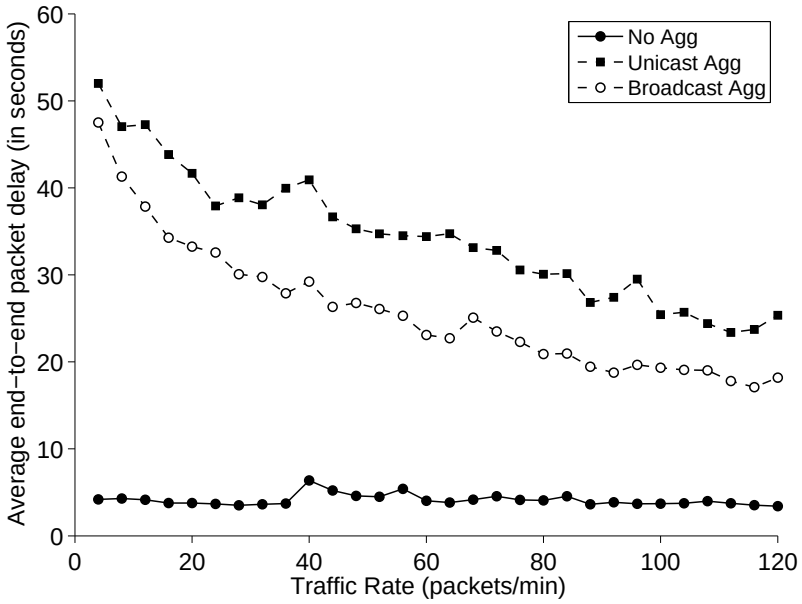


Figure 7.9. Average end-to-end packet delay

7.4.2 Delay

Fig. 7.9 displays the average end-to-end packet delay. This packet delay is measured on the individual application packets, not on the aggregated packets. We can see that with broadcast aggregation, the delay can be reduced up to 35% compared with unicast aggregation. The delay is of course higher than with no aggregation since with aggregation, packets are waiting in the queue for a certain period in order to result in fewer transmissions and save energy.

7.4.3 Throughput

In Fig. 7.10 the end-to-end throughput of the network is demonstrated. This throughput is expressed as the average number of successfully received packets per second per node. We can see that with broadcast aggregation, the throughput is increased up to 16% compared with no aggregation and up to 54% compared with unicast aggregation. This is mainly caused by faster transmission and less packet loss.

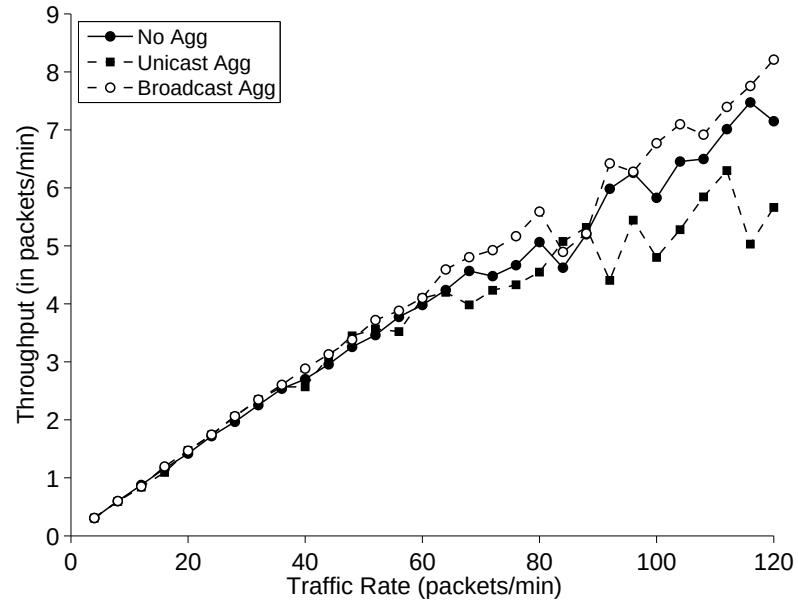


Figure 7.10. End-to-end data throughput per node

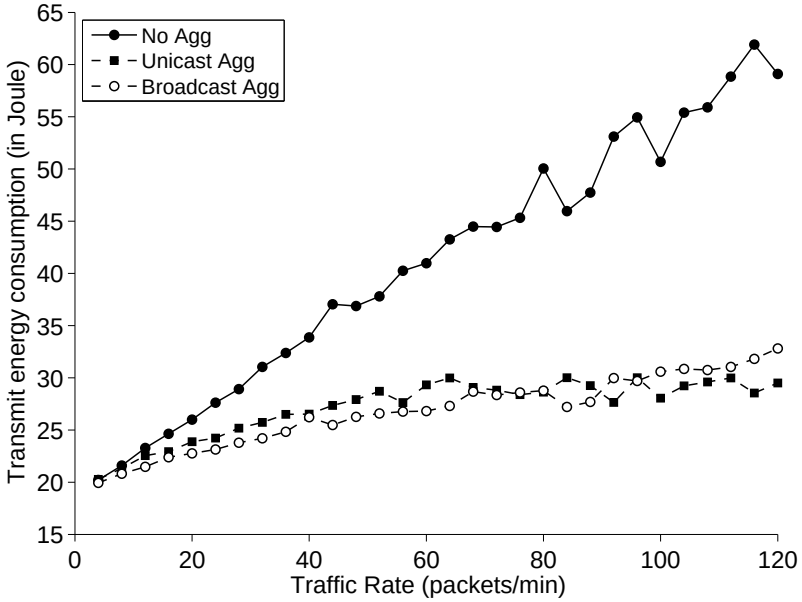


Figure 7.11. Total transmit energy consumption

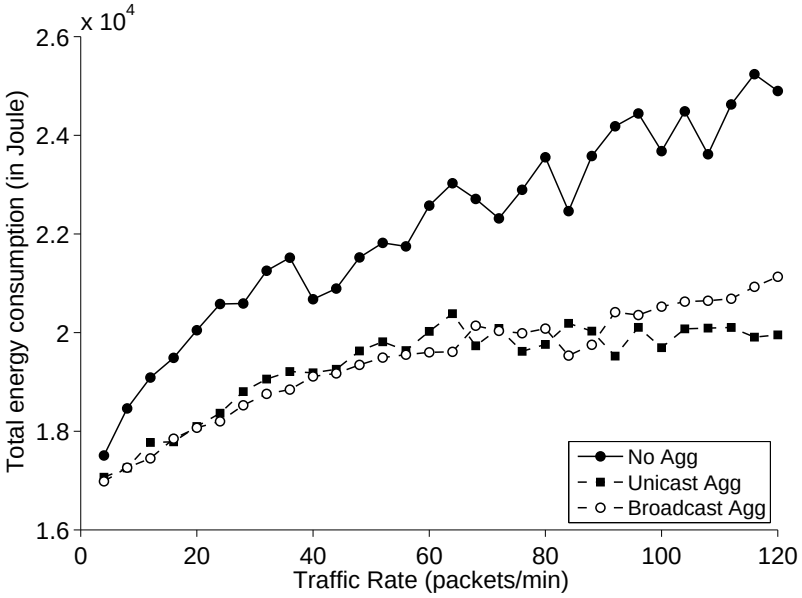


Figure 7.12. Total energy consumption

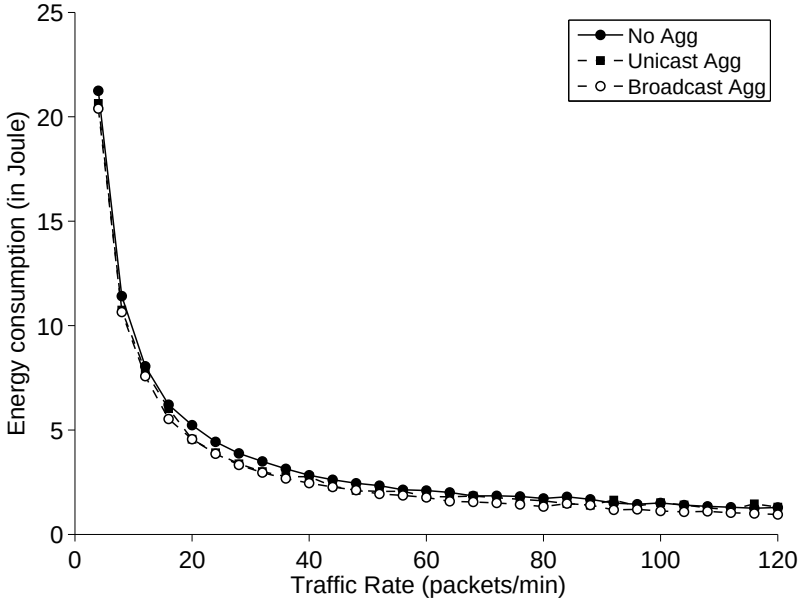


Figure 7.13. Average energy consumption per successful delivered data packet

7.4.4 Energy

Fig. 7.11 shows that the total transmit energy consumption with broadcast aggregation can be reduced up to 9% compared to unicast aggregation and up to 49% compared to no aggregation.

From Chapter 4, we know that theoretically up to 74% of energy can be saved when broadcast aggregation with $DoA_{avg} = 5$ is applied compared with no aggregation. The difference between the theoretical and the actual energy reduction can be explained by the fact that in our simulation $DoA_{avg} = 5$ is not reached. With an $DoA_{avg} = 3$, 61.75% of energy can be saved. Furthermore, we can see that for higher traffic rates, broadcast aggregation consumes more energy than unicast aggregation. This is again due to the fact that lost packets are not taken into account. Indeed, with unicast aggregation, more packets are lost and since lost packets are not further routed, less energy is consumed than expected.

Figs. 7.12 and 7.13 show the total energy consumption and the average energy consumption per successfully delivered packet respectively.

Taking into account the total energy consumption, up to 17% energy is saved compared with no aggregation and up to 4% compared with unicast aggregation. This low energy saving compared with unicast aggregation can be explained by the fact that many packets are lost when unicast aggregation is used. Since lost packets are not forwarded, nodes will have more sleep opportunities resulting in a lower total energy consumption.

When we take a look at the energy per successfully delivered packet (Fig. 7.13), we can see that up to 26% energy can be saved compared with no aggregation and up to 32% compared with unicast aggregation.

From the figures, we see that the total energy reduction gain is much lower than the total transmit energy reduction gain. The reason can be found in the used MAC protocol and can be seen in Figs. 7.14 to 7.16 which show the energy distribution of no aggregation, unicast aggregation and broadcast aggregation in terms of transmit energy, idle listening energy, receive energy and sleep energy. We can see that most energy is lost due to idle listening.

A detailed comparison of the energy contributions can be found in Figs. 7.17 to 7.18.

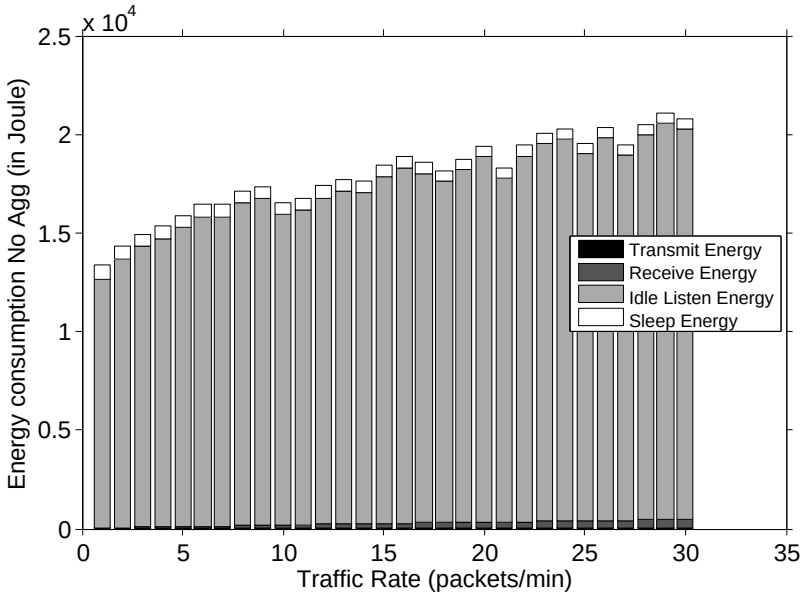


Figure 7.14. Total no aggregation energy distribution

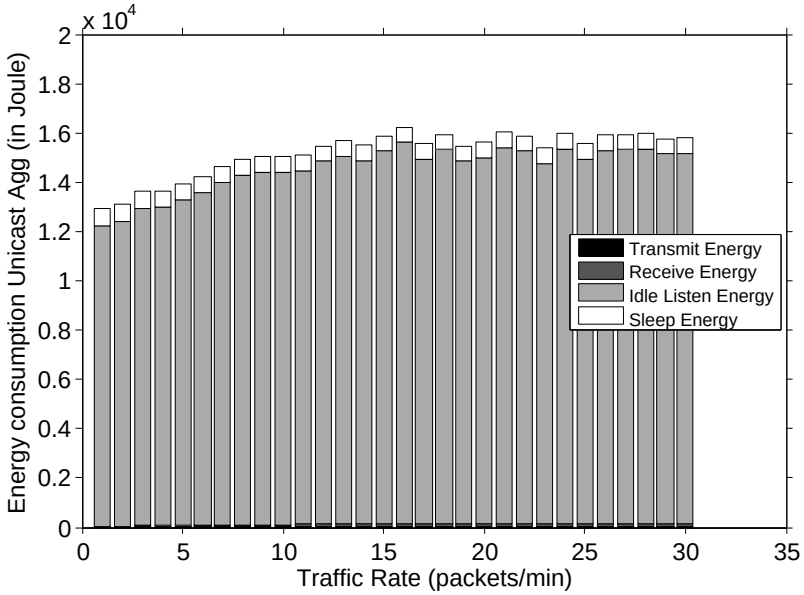


Figure 7.15. Total unicast aggregation energy distribution

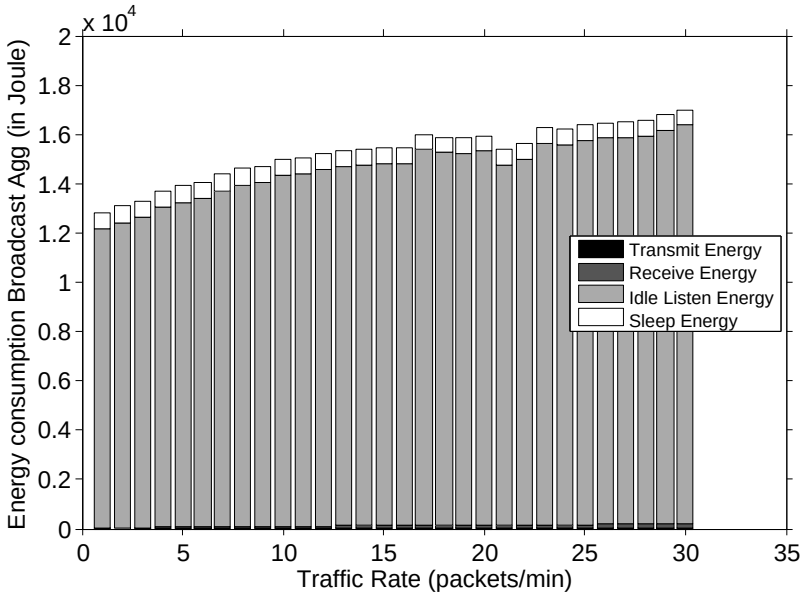


Figure 7.16. Total broadcast aggregation energy distribution

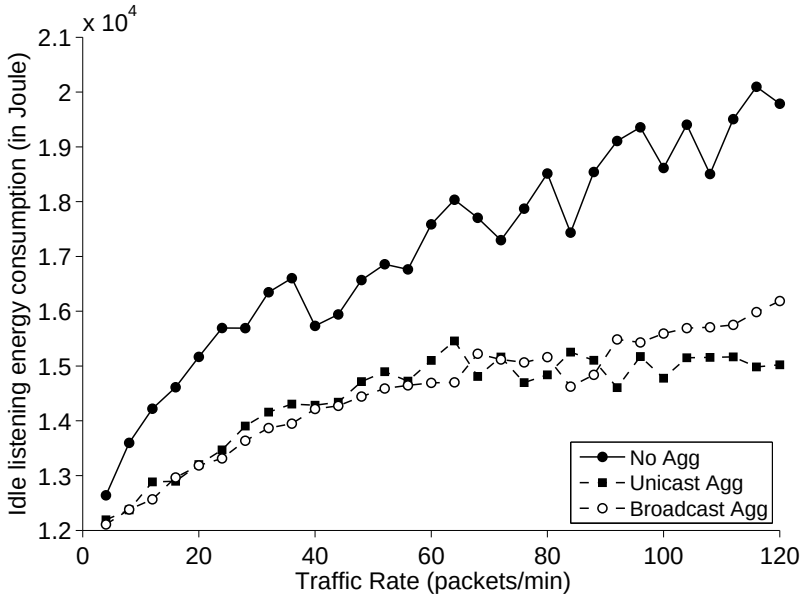


Figure 7.17. Total idle listening energy consumption

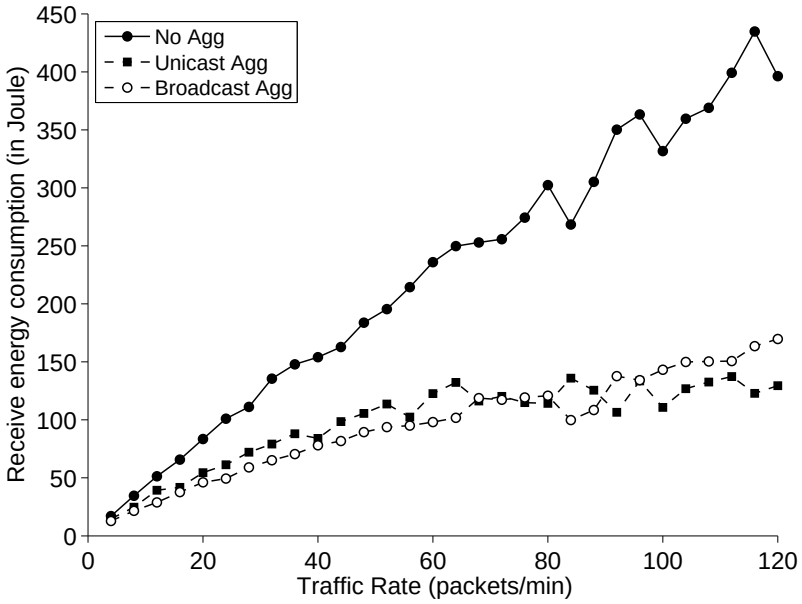


Figure 7.18. Total receive energy consumption

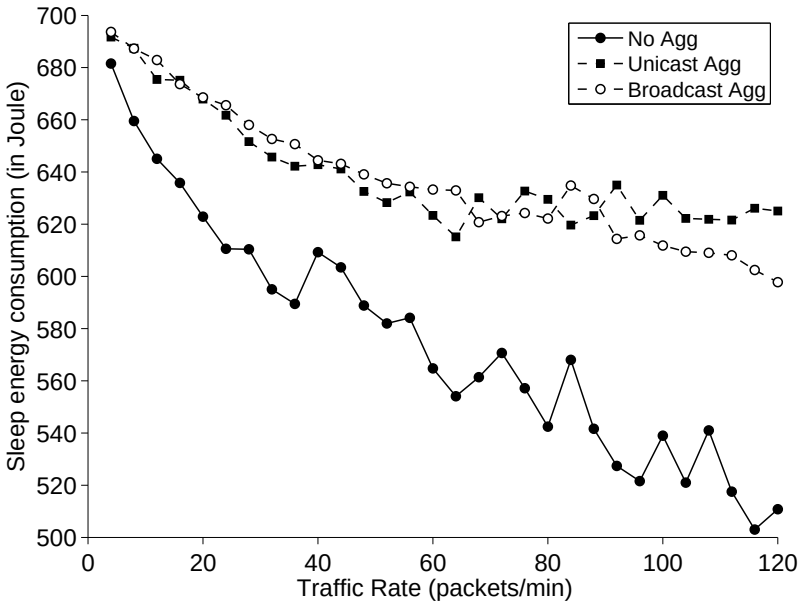


Figure 7.19. Total sleep energy consumption

Fig. 7.18 shows the total receive energy. This is the energy consumed by receiving both beacons and data, even the data that is not destined for this node.

Fig. 7.19 shows the total amount of energy that is consumed by sleeping. Both from Fig. 7.18 and Fig. 7.19, we can see that broadcast aggregation seems to perform worse than unicast aggregation for higher traffic rates. This is however not true since lost packets are not taken into account. Unicast aggregation leads to more lost packets that are not routed. This results in less receive energy consumption and more sleep energy consumption.

7.5 Conclusion

In this chapter, we investigated the use of broadcast aggregation for incoming WSN traffic. We have shown that broadcast aggregation leads to a reliability increase up to 24% compared with no aggregation and up to 37% compared with unicast aggregation. The delay can be reduced up to 35% compared with unicast aggregation and the throughput is increased up to 16% compared with no aggregation and up to 54% compared with unicast aggregation. The energy in turn is reduced up to 17% compared with no aggregation and up to 4% compared to unicast aggregation. This low energy reduction can be explained by the fact that many packets are lost. These packets are not forwarded and nodes will have more sleep opportunities and thus less energy is consumed.

References

[1] I. Ishaq, J. Hoebeke, F. Van den Abeele, I. Moerman, and P. Demeester. *Group Communication in Constrained Environments Using CoAP-based Entities*. In Proceedings of the 2013 IEEE International Conference on Distributed Computing in Sensor Systems, pages 345–350, May 2013. doi:10.1109/DCOSS.2013.14.

8 Conclusions and Perspectives

The main research question addressed in this dissertation was "How to provide Quality of Service in the next-generation wireless sensor networks that will become an essential part of the Internet of Things and that are characterized by limited resources and heterogeneous nodes, networks and applications". To this end, we have presented (i) the design of a QoS framework and (ii) the design of several QoS-aware in-network aggregation approaches. In this chapter, the most important contributions of this work are highlighted and perspectives for future research are summarized.

8.1 Contribution 1: Development of a QoS Framework for Next-Generation Wireless Sensor Networks

Conclusions:

In order to cope with Quality of Service in sensor networks with sensor nodes that suffer from limited capabilities, we have developed a QoS framework that is able to adapt itself to dynamic network topologies and heterogeneous nodes, networks and applications. The design of this framework was twofold: the architectural protocol-independent QoS approach allows a basic QoS level on nodes with limited resources, while the protocol-dependent approach allows an in-depth QoS support on nodes with more capabilities. The protocol-independent QoS ap-

proach on one hand is based on a priority level, optional QoS attributes and several packet processing rules. The protocol-dependent approach on the other hand allows tuning of protocol parameters and replacing network protocols in a distributed manner. This way, the QoS framework can adapt itself to the instantaneous QoS needs of the network.

We have shown that this framework can be implemented in any existing or future network architecture due to its modular design. To this end, we have in the first place implemented the framework as cross-layer extension of the layered Castalia network simulator. Simulation results with this framework are performed for the QoS-aware in-network aggregation approach and we have illustrated how a coupling with the SUMO toolbox can be made in order to build surrogate models for the optimization of protocol-dependent QoS parameters. In a next step, we have implemented the framework as part of the layerless IDRA sensor network architecture. We have shown that the framework allows prioritizing packet streams which can significantly reduce delay and improve reliability for high priority traffic streams.

Perspectives to testbed experiments:

During our real-life experiments, we have encountered several problems with the microcontroller and the serial stack. Learning from these experiences, IDRA is being restyled to the GUITAR framework. Moreover, new sensor nodes with increased processing and memory capacity were developed jointly by Rmoni and iMinds in the context of the ICON MoCo project [1]. Furthermore, there is a new testbed 'w- iLab.t Zwijnaarde' [2] that allows to have a better control over the environmental conditions. This testbed is located in a semi-isolated environment avoiding unwanted external interference. This makes it easier to schedule a bunch of experiments that have more or less the same environmental conditions and to avoid unpredicted outcomes of experiments, e.g. because someone turned on the microwave. Future research can investigate the QoS framework in GUITAR and experiments could be executed with the new hardware and on the new testbed. With these new tools, the experimentation results are expected to be more stable. Increased stability leads to faster collection of experimentation results and a better understanding of the behavior of the protocols.

Perspectives to simulation results:

In this PhD research, we supported protocol-dependent QoS support. This protocol-dependent QoS support was twofold: protocol parameter tuning and protocol replacement. The framework part that enables protocol parameter tuning is operational and its basic functionalities were illustrated in this thesis. In future

research, several protocols could be investigated in-depth and can be optimized. Such analysis could be done both on individual network protocols, but also on cross-layer protocol interaction. Furthermore, dynamic protocol replacement is also an interesting topic that can be investigated in more detail in future work. Currently, each protocol has often its own specific target traffic and network operation area. In future research, for each of these different protocols the optimal working area can be determined. With dynamic protocol replacement, the network protocols can then be replaced by a more appropriate protocol.

8.2 Contribution 2: Development of QoS-aware In-Network Aggregation Protocol Approaches for Next-Generation Wireless Sensor Networks

Conclusions:

We have seen that in-network aggregation is a technique that is often used in wireless sensor networks to reduce energy consumption. However, we have shown that this could have a negative impact on QoS metrics such as delay and reliability. Therefore, we investigated how in-network aggregation could be made QoS-aware.

First, we showed that our QoS frameworks is very suited to support QoS-aware in-network aggregation due to the shared queue that allows a global node view and a load-balanced storage.

Afterwards, we presented three QoS-aware in-network aggregation approaches for sensor network traffic that is interconnected with the Internet of Things: one for outgoing WSN traffic (towards the IoT), one for local WSN traffic, and one for incoming WSN traffic (from the IoT).

Firstly, for the outgoing WSN traffic, we presented a unicast-based protocol-independent tunable QoS-aware in-network aggregation scheme. This approach allows making an optimal trade-off between energy reduction and QoS requirements. We have shown that reliability can be increased up to 16% while the delay remains within the given limits and while there is an overall energy reduction up to 19%.

Secondly, for the local WSN traffic, we proposed to use broadcast aggregation instead of unicast aggregation as solution to overcome delay, reliability and energy issues in networks with fewer aggregation possibilities. When queues become filled with packets for many different destinations, there are 2 options: drop packets or send packets with fewer aggregated packets into it. The former leads to a decreased reliability while the latter leads to more energy consumption. When broadcast aggregation is used, packets are sent independently of their next-hop destination by broadcast, which releases the burden. We have shown that the queue

occupation was reduced on average with 2 (out of 15) places. This led to an increase in throughput and reliability up to 23% compared with no aggregation and up to 15% compared with unicast aggregation. Furthermore, the average queue delay was decreased by 52% compared with unicast aggregation. Finally, broadcast aggregation led to an energy reduction up to 27% compared with no aggregation and up to 13% compared with unicast aggregation.

Finally, we applied the broadcast aggregation approach to incoming WSN traffic and we have shown that broadcast aggregation leads to a reliability increase up to 24% compared with no aggregation and up to 37% compared with unicast aggregation. The delay could be reduced up to 35% compared with unicast aggregation and the throughput was increased up to 16% compared with no aggregation and up to 54% compared with unicast aggregation. The energy on its turn was reduced up to 17% compared with no aggregation and up to 4% compared with unicast aggregation.

Perspectives:

We have observed that the performance gain by applying unicast and broadcast aggregation optimization strategies strongly depends on the used network protocol (especially the MAC protocol). Therefore, the impact of the used MAC and routing protocol on the simulation results can be investigated. The MAC protocol used in our simulations suffers from a high idle listening time. One optimization could be to add the destination address in the MAC broadcast message in order to prevent nodes to stay awake unnecessarily. Furthermore, the impact of a real environment on the proposed aggregation strategies can be investigated. In a realistic environment, different sources of interference may be present (e.g. WiFi traffic). Interference typically results in more unreliable links and, consequently, increased packet loss. Furthermore, bigger packets, such as those obtained after aggregation, are more vulnerable to interference compared to smaller packets. To achieve high reliability in the presence of packet loss, retransmissions are needed, something that is harder to achieve for broadcast traffic as there are multiple receivers of the same packet. It is clear that the aggregation results will become less predictable and that a trade-off between broadcast and unicast aggregation may exist depending on the particular characteristics of the real environment.

Another topic that could be investigated in future research is the impact of different traffic streams on the provided QoS level. It would be very interesting to investigate the behavior of these packets since they can be aggregated together with other (non-prioritized or with another priority) packets having different delay and reliability constraints.

References

[1] *RM090*. http://www.crew-project.eu/portal/wilab/rm090/. [Online; accessed 17 March 2014].

[2] *w-iLab.t Zwijnaarde*. http://www.crew-project.eu/portal/wilab/open-environment-testbed-zwijnaarde. [Online; accessed 17 March 2014].