



FP7-ICT-SEC-2007-1
Contract no.: 225186
www.wsan4cip.eu



WSAN4CIP

Deliverable D1 . 3

Modular design and performance ranking of communication protocols

Editor:	Evgeny Osipov
Deliverable nature:	Report (R)
Dissemination level: (Confidentiality)	Public (PU)
Contractual delivery date:	31 December 2009
Actual delivery date:	31 December 2009
Suggested readers:	Research community
Version:	1.0
Total number of pages:	29
Keywords:	performance ranking, modular design, MAC protocols.

Abstract

In this deliverable we present a systematic approach towards designing modularized protocols and rank a contribution of their components to the overall system performance. In the nutshell, this approach is based on three steps: 1.) identifying adjustable parameters in existing protocols, 2.) ranking their influence on the system-level performance metrics and 3.) defining protocol modules exposing the parameters of the highest rank. To this end we present the definition of the components for constructing MAC protocols based on ranking of the impact of adjustable parameters on the overall system performance. We also overview a ranking method for functional blocks of protocols on the routing layer.

Disclaimer

This document contains material, which is the copyright of certain WSAN4CIP consortium parties, and may not be reproduced or copied without permission.

In case of Public (PU):

All WSAN4CIP consortium parties have agreed to full publication of this document.

Impressum

Wireless Sensor Networks for Protection of Critical Infrastructures

WSAN4CIP

WP1 “CIP requirements and methodologies”

Modular design and performance ranking of communication protocols

[Editor: Name, company] Evgeny Osipov, Luleå University of Technology

Copyright notice

©2009 Participants in project WSAN4CIP

Executive summary

The overall goal of work package WP1 “CIP requirements and methodologies” is to make engineering of dependable systems more systematic. The engineering of dependable systems is a multidimensional process. On the one hand a particular application dictates its performance requirements essential to fulfill its functional goals. On the other hand a particular installation site impose a set of factors limiting the performance of the network protocols. Finally the stack of communication protocols expose a set of adjustable parameters on different communication layers for tuning the overall performance of the system so that it satisfies the requirements of the applications in the given operating environment.

Currently there exist dozens of application-specific protocols on all layers of the communication architecture. All current protocols are known as monolithic pieces of software. The fundamental question to answer in this respect is which protocol is best suited for a specific application and operating environment? This question would be easy to answer given a stable ranking system of different protocols with respect to their contribution to the overall system performance. A rather straightforward way of constructing such a ranking system would be to benchmark performances of different protocols in different communication contexts. Most of the existing results reported in the literature are produced using this approach. The major shortcoming of such analysis, however, is that in some specific scenarios while a particular protocol would demonstrate better performance, this performance is still not optimal for the considered operating environment. This is because it is not a particular protocol as an inseparable unit which in cooperation with other inseparable protocols produce one or another performance picture. It is the specific protocol’s component (which may be present in several other protocols) and its particular parametrization for a given scenario that produces one or another impact on the overall system performance. An optimal combination of the components for a specific scenario may not be present in any existing monolithic protocol, and therefore will constitute a completely new protocol.

The goal of this deliverable is to present a systematic approach towards designing modularized protocols and ranking contribution of their components to the overall system performance. In the nutshell, this approach is based on three steps: 1.) identifying adjustable parameters in existing protocols, 2.) ranking their influence on the system-level performance metrics and 3.) defining protocol modules exposing the parameters of the highest rank.

To this end we present the definition of the components for constructing MAC protocols based on ranking of the impact of adjustable parameters on the overall system performance. We also overview a ranking method for functional blocks of protocols on the routing layer. It is important to note, that the task of modularization (even of one communication layer) is complex and requires substantially more efforts beyond those planned for work package WP1. The work described in this document, therefore, will be taken over as part of work package WP3 activities along the lines of identifying new functional components and further ranking of the impact of the their adjustable parameters on the overall system performance.

List of authors

Company	Author
LTU	Wolfgang Birk
LTU	Evgeny Osipov
LTU	Laurynas Riliskis
NEC	Alban Hessler

Contents

Executive summary	3
List of authors	4
1 Introduction	8
2 On modularized protocol design	8
3 Ranking of functional components on routing layer	9
3.1 Definition of route maintenance modules for a reactive routing protocol	9
3.2 Ranking of route maintenance components	9
3.2.1 AODV-LL	10
3.2.2 AODV-HELLO	10
3.2.3 tinyLUNAR	10
3.2.4 Opportunistic LUNAR	10
3.3 Summary and ranking	10
4 Related Work	11
4.1 Modular approaches to the design of network protocols	11
5 Performance Analysis and Definition of Critical Parameters	12
5.1 Performance metrics	12
5.2 Experimental setup	13
5.3 Individual tests	16
5.3.1 Latency test	16
5.3.2 Throughput test	16
5.4 Analysis Methodology	17
5.5 Results	17
6 Modularization of MAC layer	20
6.1 Components of Medium Access Control protocol	20
6.2 From functional patterns to components	21
6.3 Classification and formal representation of components	21
6.4 Analytical validation of component based modeling	21
7 Discussion and future work	24
8 Summary of Deliverable 1.3	25

List of Figures

1	Definition of the experimental data.	15
2	Testbed Topology	16
3	Latency Test	17
4	Throughput Test	17
5	Time series plot of the min-max scaled data sets. Rows related to the performance metrics and columns to the parameters.	18
6	Scatter plot of the data sets. Rows related to the performance metrics and columns to the parameters.	19
7	Component based MAC.	22
8	Simplified tree ontologies for MAC components. By bold lines the nodes needed for presenting an example of composition are shown.	24
9	Derivation of the control law T_{TX} for MAC instance 1. The left branch indicated by dashed arrows shows compositions of the channel manager and the right branch of composition of the message exchange component.	25

List of Tables

1	Identified adjustable parameters on different communication layers.	13
2	IEEE 802.15.4 radio based Sensor Platforms.	14
3	Correlation coefficients	18
4	Description of selected components, their parameters and properties.	23
5	Parameter description for XMAC and BMAC in our model.	24

1 Introduction

Systematic engineering of dependable systems is a multidimensional problem. On the one hand a particular CIP application dictates its performance requirements essential to fulfill its functional goals. On the other hand a particular installation site imposes a set of factors limiting the performance of the network protocols. The ability of a system to deliver the specified level of services, is the major requirement on the communication architecture for most applications of wireless sensor networks (WSN). The large variety of WSN applications makes, however, it challenging if not impossible at all to design a universal set of protocols which would deliver specified quality of service to all applications. This conclusion while being well understood in the research community [26], in reality is visible by a great number of application specific and in many cases hardware dependent protocols implementing MAC, routing, transport functionality. Now, given a set of existing protocols on different communication layers and new application and operating environment which one the developer should choose to fulfill the performance requirements of the application?

This question would be easy to answer given an established ranking system of different protocols with respect to their contribution to the overall system performance. A rather straightforward way of constructing such a ranking system is to compare performances of different protocols in different communication contexts. Most of the existing results reported in the literature are produced using this approach. The major shortcoming of such analysis is that in some specific scenarios while a particular protocol would demonstrate better performance, this performance is still not optimal for the considered operating environment. This is because it is not a particular protocol as an inseparable unit which in cooperation with other inseparable protocols produce one or another performance picture. It is the specific protocol's component (which may be present in several other protocols) and its particular parametrization for a given scenario that produces one or another impact on the overall system performance. An optimal combination of the components for a specific scenario may not be present in any existing monolithic protocol, and therefore will constitute a completely new protocol.

Being able to combine functional components into an application-specific stack of protocols (or in other words having a modularized structure of the protocols) is the essential prerequisite towards optimization of WSN performance. As a matter of fact modularized protocols do not exist presently.

The goal of this deliverable is to present a systematic approach towards designing modularized protocols. In the nutshell, this approach is based on three steps: 1.) identifying adjustable parameters in existing protocols, 2.) ranking their influence on the system-level performance metrics and 3.) defining protocol modules exposing the parameters of the highest rank.

In this document we present the details of modularization of MAC protocols, as the first software block in the communication stack. Being such, MAC layer exposes the least degree of cross-layer dependencies from lower layer functionality, therefore, it is the correct starting point for the design of modularized protocols. As for the protocols on higher layers, i.e. routing and transport, we overview a ranking method of functional blocks from our previous work. It is important to note, that the task of modularization is complex and requires substantial efforts. The work described in this document, therefore, will be continued in other work packages along the lines of identifying new functional components and further ranking of the impact of the their adjustable parameters on the overall system performance.

This document is organized as follows. We present a motivation for modularized design of communication protocols in Section 2. Section 3 presents the evaluation of route maintenance components impact on the system level performance. Section 4 describes the related work in the domains of ranking the influence of functional components on routing and transport layers, methods for performing the ranking analysis and finally efforts on modularizing protocols. In Section 5 we introduce system level performance metrics and the ranking analysis of the adjustable parameters on MAC layer. Further in Section 6 we use these results to identify components of MAC protocols and demonstrate the technique of combining the reusable components into different protocols, known presently as monolithic solutions. The discussion and the outline of the future work is presented in Section 7. We conclude the deliverable in Section 8.

2 On modularized protocol design

Flexibility of tailoring a protocol stack for a particular application without the need for rewriting the code is the major motivation behind modular design of protocols. It was recognized already in the beginning of 1980s

[13] that it is difficult to improve the performance of the already existing program. The performance of the protocols is influenced by all included functional blocks which may behave differently in different communication contexts. In many cases it is necessary to redo the protocol implementation completely which requires significant efforts and skills from the protocol developers.

As an example one could imagine a general purpose reliable transport protocol which was equipped with all sorts of special mechanisms by which it would communicate with protocols on other communication layers and modify its behavior according to the specific conditions of the operating environment. On the other hand imagine not one but several protocols which could be combined by substituting different functional blocks into a common protocol core. This structure has the disadvantage that there will be several possible instances which needed to be verified on logical and performance correctness. Developing a protocol having in mind both logical correctness and performance is recognized extremely difficult task for the developers.

This understanding resulted in the situation that while the ideas of modularized protocols are circulating in the research community for more than two decades now, implementations of particular protocols using this approach *do not exist presently* [14]. This statement is valid both for the traditional Internet protocols and protocols for wireless sensor networks in particular. The later class of communication systems offers, however, an excellent playground for resurrecting the concept. The absence of the stable system architecture for wireless sensor networks, a clear need for tailoring the content of the communication stack to the specifics of particular application, and finally very computationally limited hardware platform for sensor nodes preventing from the implementation of heavy-weighted universal solutions indicate clear promises for the success of the component-based protocol design approach in reality.

3 Ranking of functional components on routing layer

In this section we consider ranking of two major types of invocation of the broadcast control traffic generated during route maintenance phase of reactive routing protocols: 1) Error-driven and 2) periodic. We take two protocols, originally designed for mobile ad hoc networks and later adopted for application in wireless sensor networks: AODV [40] and TinyLUNAR [38]. The ranking of routing components presented in this section is based on the results in [39].

3.1 Definition of route maintenance modules for a reactive routing protocol

From the two considered protocols tinyLUNAR exhibits modular structure, where it is possible to customize an address resolution module, select different path maintenance patterns. The current implementation of TinyLUNAR deploy two different route maintenance patterns. The first pattern represents periodic invocation of broadcast route requests every 3 seconds. This allows to eliminate the complexity of maintaining routing state in intermediate sensor nodes. The second variant tailor invocation of route requests to the actual activities of ongoing data sessions. As long as there are packets generated by the source sensor the route re-establishment timer is rescheduled three seconds in the future.

In order to analyze other operational patterns, which are not implemented by tinyLUNAR we selected another popular protocol of the same class AODV. We considered two variants of this protocol. In the first variant the HELLO mechanism is enabled to maintain the connectivity between neighbors further on we refer to this variant of AODV as to AODV-HELLO. In the second AODV variant, further referred as to AODV-LL, the route maintenance is done by means of explicit link layer feedback. In the former case the loss of connectivity between nodes forwarding traffic of a specific connection is detected by three missing HELLO messages; in this case the route maintenance procedure is invoked and the problem is signaled back to corresponding sources. In AODV-LL a packet loss during the transmission is detected by the link layer; the problem is then immediately reported to the routing engine, which in turn invokes the route maintenance operations.

3.2 Ranking of route maintenance components

For the ranking method TCP traffic was used to excite the network. TCP fairness was used as a system level performance metric to analyze the impact of routing components. TCP flows were calibrated such that in the absence of routing a perfect fairness is achieved. When routing protocols were activated the deviation from the perfect fairness indicated the degree of influence of route maintenance patterns on the overall performance.

The details of the experiments including simulation setup as well as performance plots could be found in [39]. Here we reflect the major findings.

3.2.1 AODV-LL

The least impact on the ideal TCP behavior is introduced by the routing traffic pattern of AODV with link layer feedback enabled. This is because of the error-driven invocations of broadcast transmissions. In AODV-LL the broadcast activities are initiated as a reaction to packet losses reported by the link layer to the AODV engine. In the case of a packet loss the protocol assumes that the connectivity to the corresponding neighbor is lost and initiates the route recovery procedure. In networks of up to 30 nodes, the broadcast traffic does not introduce enough overhead to force TCP flows into the routing induced slow start phase. However, beyond 30 nodes the broadcast bursts caused by every lost packet are long enough to cause the invocation of the slow start phase at a TCP sender; this leads to stammering TCP flow progress and worse fairness figures.

3.2.2 AODV-HELLO

The most unstable effect on the quality of TCP sessions is introduced by the traffic patterns produced by AODV-HELLO. This is due to uncoordinated transmissions of HELLO messages. Despite of the small size, their frequent and independent emissions from multiple nodes does not allow any of the competing TCP sessions to progress smoothly through the network.

3.2.3 tinyLUNAR

As for the effect of the routing traffic pattern generated by tinyLUNAR, we observe that it is very similar to that of AODV-HELLO. If in the case of AODV-HELLO we have short frequent one hop broadcasts, in the case of LUNAR we have less frequent but more massive flooding waves, which result in stammering TCP progress with increasing number of nodes in the network. We conclude that complete route rediscovery is not suitable for networks with more than 15 densely distributed nodes.

3.2.4 Opportunistic LUNAR

When changed the route refresh strategy of tinyLUNAR to one which closely resembles the error-driven pattern of AODV-LL, the influence of route maintenance component reduced dramatically. The modification concerned disabling the forced three second complete route rediscovery mechanism at sources. Instead, we retain a route as long as there are packets arriving to the forwarding engine: With every new data packet we shift the route timeout three seconds into the future. This modification allowed us to create an opportunistic version of LUNAR.

3.3 Summary and ranking

In summary the error-based route maintenance components are suitable for networks up to 30 - 50 nodes in the bottleneck region. Hello-based component and forced complete route rediscovery are suitable for substantially smaller networks of up to a dozen nodes in the bottleneck region. In networks of such scale built of extremely computationally limited nodes, however, periodic route maintenance components may be more beneficial due to a simpler implementation. We highlight again that these conclusions are drawn for a specific class of applications which require reliable transport and window-based congestion control functionality. We conjecture that following this approach of benchmarking components influence by testing all possible variants in all possible scenarios is extremely time-consuming task. Moreover, not all operating environments could be foreseen during such analysis. This conclusion calls for new methods of joint definition of modularized protocols along with the formal framework for their analysis. We describe the results of our work in this direction on the example of designing a modular MAC protocol.

4 Related Work

When it comes to the analysis of the performance of protocols, the assessment is conducted for certain applications, traffic patterns or with a performance metrics in mind. As a result, the outcome is hard to compare and it is difficult to determine a-priori if a protocol is feasible for a deviating application or topology. Some examples are [49, 1, 30].

A different approach to analyze and tailor protocols comes from the optimization and automatic control community, where mathematical frameworks for network architectures are sought. Within the subject area of *Networked Control Systems*, three main themes are of interest: Control of networks, control over networks and multi-agent systems. A summary of the current trends and accomplishments in the subject area are found in [50]. Within the scope of this work the first theme is of interest and since the late 1990s, efforts on flow control for various internet protocols based on optimization schemes is addressed. Commonly understood are the facts that network dynamics and protocol implementations interact, as well as the presence of time delays in the feedback path.

In [50] and [42] it is also indicated that the design of network protocols should be based on a model of the underlying network together with a specification of the application requirements. There it is also advocated that a optimization based approach in a distributed setting would yield the best results. One of the main conclusions for wireless networks is the need for optimization scheme that considers several protocol layers and that control theoretic tools are needed for the analysis of protocol dynamics. As a result, mathematical frameworks are under development and the application of optimization schemes is initiated, see [10, 11, 25, 12].

Specific examples of the ranking of different components of routing and transport layer protocols are [39, 9]. In the first article the authors analyze the influence of the presence of the link layer feedback and different route maintenance techniques on the throughput of transport protocol with window-based congestion control. It is shown that protocols with the route maintenance components which are aware of the actual events of message loss place significantly lower impact on the application-level throughput than link-layer agnostic. The second paper focuses on the ranking of the different parameters of the congestion control component in TCP-like reliable transport protocol on the application level throughput. Amongst the most important findings is that the congestion control component should be ranked jointly with routing components.

4.1 Modular approaches to the design of network protocols

A component based development (CBD) approach to design of complex software systems was suggested back in nineteen sixties [37]. Application of the component based paradigm to the design of network protocols was not in question until the beginning of nineties. By this time the layered model was a standard development approach. Inflexibility of the architecture was recognized in number of papers, e.g. [23, 43]. Incompatibility problems between protocol stacks from different vendors and inability of the system to dynamically reconfigure its functionality across the layers called for alternative component based design approaches.

In wireless networks the problems with layered design have only exacerbated. The need for cross layer adaptation for better protocol performance in heterogeneous communication contexts and inability of doing so systematically across the hard layer boundaries call for new design paradigm [27].

The CBD approach to the development of software in wireless sensor networks was adopted more or less from the time of their appearance on the technology arena [21]. In addition to the simplicity of application development due to straightforward abstractions for programmers, one of the major motivating factors for adopting CBD was the *efficiency of the resulting code base*. Efficient code base is achieved by exclusion of unnecessary, for a particular application, components. A straightforward extension of the paradigm to the design of communication protocols [28] aimed at *reducing hardware dependency* of particular protocols.

A work that closely relates to some ideas of the performance driven CB-design of WSN considered here is [35]. The authors outline a methodology for metadata specification and metamodel for description of network components and network configurations. Further the protocols are viewed as an entity, component. In our work we model the protocol itself, based on requirements and assessed environmental model.

The CB-design of MAC protocols for wireless medium have been implemented and tested in MultiMAC [18]. This work extends SoftMAC [44, 46, 45] and is based on MetaMAC [20] concept. In MetaMAC authors address the problem of choosing the right MAC protocol for the specific scenario. The proposed framework implements a higher layer of adaptivity, on top of existing MAC protocols. The selection of MAC protocol

to be used is done from the available network performance information. Therefore, the selection of the MAC protocol is transparent. The MultiMAC utilize this approach by implementing TDMA, Cheesy, Aloha, 802.11 MAC protocols. MetaMAC, and so the implementation, MultiMAC acts as middleware between PHY layer and existing MAC implementations. Based on the current network metrics MultiMAC select the best MAC layer for the best performance. Even if the main goal of selecting a MAC protocol with best performance for a given scenario is the same in ours and the MultiMAC framework, the postulation for achieving it is different. MultiMAC and MetaMAC is suitable for wireless networks when the computational power and the energy resources allow its implementation and operation. In the case of WSN the size of the runtime code should be extremely compact and the overall operation should be energy efficient. In our framework we propose a synthesis of a MAC protocol from existing generic components.

Requirements-based programming has been proposed in [22]. The authors provide a framework for expressing requirements as a set of services which a WSN provides. This allows to develop a system with high reliability which correctness can be proved. Further it allows to ensure that the actual code is consistent with system requirements and meets desirable level of performance.

5 Performance Analysis and Definition of Critical Parameters

A priori performance analysis reduces the design cycle for a WSN and enables a more systematic network protocol design and commissioning. The performance analysis is used to rank functional components of a protocol for their suitability and adaptability to a specific scenario. Thus, ranking is not aiming at benchmarking of different protocols in relation to each other. Instead, the aim is to determine if a given protocol provides the necessary properties to realize a WSN scenario according to its specification.

The scenario specification, in turn, provides information and requirements on:

- Desired application;
- Traffic pattern;
- Layout and topology;
- Performance characteristics.

The first three items form the boundary conditions for the performance analysis and the last item is used to rank the outcome of the analysis for a specific protocol. A quantification of the outcome is achieved using performance metrics that relate to the required performance characteristics. The metrics are determined prior to the performance analysis and will affect the setup of experiments and simulation studies. Beside an indication of the suitability of a protocol the significance of protocol parameters on the determined performance metrics is quantified and can be used in a subsequent modularization of the selected protocol.

5.1 Performance metrics

The set of system-level performance metrics is in fact well defined in the domain of the wired Internet. The known metrics of network performance to be used by network operators, end users or independent testing groups include *connectivity* [33], *delay* and *loss* [2, 3, 4], *delay variation* [16], *loss patterns* [31], *packet re-ordering* [36], *throughput* (or bulk transfer capacity [34]), energy consumption index, fairness index, packet delivery ratio. These metrics can further be mapped into a higher-level performance metrics meaningful for end-users of the network (i.e humans), for example reliability level, mean opinion score etc. Applications of different types place different requirements either on the whole set of metrics or its subset. For example, an application prioritizing high data reliability over other characteristics would demand maximization of end-to-end delivery ration even by the cost of reduced life time due to non-optimal energy consumption and other performance characteristics. In this document we, however, do not further discuss the tradeoffs of simultaneous optimization of different system-level performance metrics. Instead we are seeking an answer on a question to which degree an in what direction (increase or decrease) the particular system-level performance metric will change. For our experiments we selected the following system level performance metrics most suitable for wireless sensor networks:

Layer	Parameters	Affected performance metrics
Transport	RTO - retransmission timer Size of retransmitted block (congestion or (re-)transmission window) Retransmission strategy	end-to-end throughput End-to-end delivery ratio EDR, throughput, fairness
Routing	hop limit route age route timeout route maintenance strategy	Throughput, EDR, ECD end-to-end delay
MAC	Maximum payload Transmission power CCA length CCA level preamble length back-off time	Throughput, EDR E2E delay, ECD

Table 1: Identified adjustable parameters on different communication layers.

- End-to-End delay;
- Throughput;
- End-to-end delivery ratio (EDR);
- Energy consumption index (ECI);
- Fairness index.

In this deliverable we evaluate the impact of communication components and their adjustable parameters with respect to end-to-end delay, throughput and end-to-end delivery ratio metrics. The analysis of the energy consumption index and the fairness index will be reported as results of our future work.

The performance of the particular WSN application is shaped by joint functionality of transport, routing and MAC protocols. Protocols on each layer expose a set of adjustable parameters. In Table 1 we identify these adjustable parameters. Note that at this point this list of parameters is not fully complete. The goal for this deliverable was to identify a limited set of essential parameters in order to start the development of the analysis framework. As soon as the stable operation of the framework will be achieved this list might be extended with more parameters.

It is important to note that adjustment of the protocol parameters listed above is meaningful only within the context of particular environmental characteristics. These characteristics constitute another type of parameters one has to account for in the optimization process. We call these parameters environmental parameters and identify the following:

- Distance - the distance between a sender and a receiver;
- Directionality of the antenna;
- Height of the antenna;
- Traffic pattern;
- Network topology.

5.2 Experimental setup

The goal of the experiments is to identify the response of system-level metrics on changes in the identified adjustable parameters. At first we performed experiments only with adjustable parameters of MAC protocols.

Those are summarized in Table 1. For each parameter we define its operating range through minimum, maximum and resolution of the parameter values. The parameter values for a parameter P^i during the experiment need to be within the operating range and is given by

$$P^i(t) = P_0^i(t) + n^i(t)$$

There, a $P_0(t)$ is the nominal value and $n(t)$ is white noise with a predetermined variance. Although P_0 can be constant behaviours like steps and ramps can be exhibited. Since the communication process has a multi-variable (MIMO) character, it is necessary to design the parameter values for the experiments such that all parts of the process are excited. This requires that the noise sequences for different parameters P^i , P^j are uncorrelated and that changes in the nominal values occur at different times.

Since the team that conducts the experiment and the one which performs the analysis are located in different geographic locations, there is a need to communicate the experimental setup and the parameter values for the experiment in way that offers flexibility during the experiment runs and still offering fulfillment of the above mentioned requirements.

The experimental data is therefore defined in an Excel data sheet. For each parameter there are two double columns with values. The suggested nominal values and variations around the nominal values are predetermined and provided to the test operators. Prior or during the test the operator has the possibility to adjust the parameter values using a gain factor α and offset β for both nominal values and variations. The resulting parameter values are therefore determined by

$$P^i(t) = (\alpha_0 \cdot P_0^i(t) + \beta_0) + (\alpha_n \cdot n^i(t) + \beta_n)$$

The resulting parameter values for the experiment are automatically calculated by the data sheet. Figure 1 shows a partial screen shot of the data sheet. This procedure yields a freedom for the test operator while not violating the test requirements.

Due to the limitations of current simulators and emulators we opted for a real testbed to run our experiments. We started with a small, limited setup in order to have better control on the network behavior and to increase the accessibility of errors and potential improvements in the setup. Therefore, we will only present first results from this initial setup in this milestone. In a next step the testbed will be scaled up to a more ambitious and realistic network environment.

The testbed for the experiments comprises a set of TelosB nodes, with hardware characteristics according to Table 2.

	CPU	RAM	ROM	Radio	Flash
MicaZ	Atmega128 (8 bits)	4 kB	128 kB	CC2240	1 MB
TelosB	MSP430 (16 bits)	10 kB	48 kB	CC2240	512 kB

Table 2: IEEE 802.15.4 radio based Sensor Platforms.

The experiment itself consists of three distinctive parts: setup, measurements, and results analysis.

1. Setup

- (a) Select parameters input and the desired performance parameters to measure.
- (b) Generate for n tests a value for each input parameter (see a list on Table 1). These values are the basis for the measurements and are stored in a table sheet such as Excel.

2. Measurements

- (a) A possible testbed setup is depicted on Fig 2. It is as least composed by one master controller which orchestrates the different test deployments. There might be some slave computers, which are used to easily spawn the test over a large area by using the existing infrastructure, such as the wired or wireless LAN.
- (b) The master configures a particular test. It first imports the table sheet that was generated in the setup phase, then for each tuple of input parameters, the master:

Parameter	CCALength.Var Time during which the node will check if the channel is clear duty cycle length ms	CCALevel.Nom Energy level above which the node will consider the channel to be clear dBm	CCALevel.Var Energy level above which the node will consider the channel to be clear dBm	PreambleLength.Nom The length of the preamble message byte	PreambleLength.Var The length of the preamble message byte	Back-OffTime.Nom Back-Off length when sending symbol (half a byte)	Back-OffTime.Var Back-Off length when sending symbols (half a byte)
Excitation	1	-96	-96	2	2	20	20
	0	-65	0	16	16	INF	INF
Excitation	1	-45	-8	1	1	1	1
	0	-65	0	1	1	1	1
Excitation	2.068209925	3.793216209	3.661387086	1.137964863	1.331471379	7.586432418	3.392271005
	2.068209925	3.793216209	3.661387086	1.137964863	1.331471379	7.586432418	3.392271005
Excitation	5	-10	-8	3	2	20	6
	-5	-10	-8	-3	2	20	6
Excitation	5	-75	10	11	2	70	6
	-5	-75	10	11	2	70	6
Excitation	2.068209925	3.793216209	3.661387086	1.137964863	1.331471379	7.586432418	3.392271005
	2.068209925	3.793216209	3.661387086	1.137964863	1.331471379	7.586432418	3.392271005
Excitation	5	-10	-8	3	2	20	6
	-5	-10	-8	-3	2	20	6
Excitation	5	-75	10	11	2	70	6
	-5	-75	10	11	2	70	6
Excitation	2.068209925	3.793216209	3.661387086	1.137964863	1.331471379	7.586432418	3.392271005
	2.068209925	3.793216209	3.661387086	1.137964863	1.331471379	7.586432418	3.392271005

Figure 1: Definition of the experimental data.

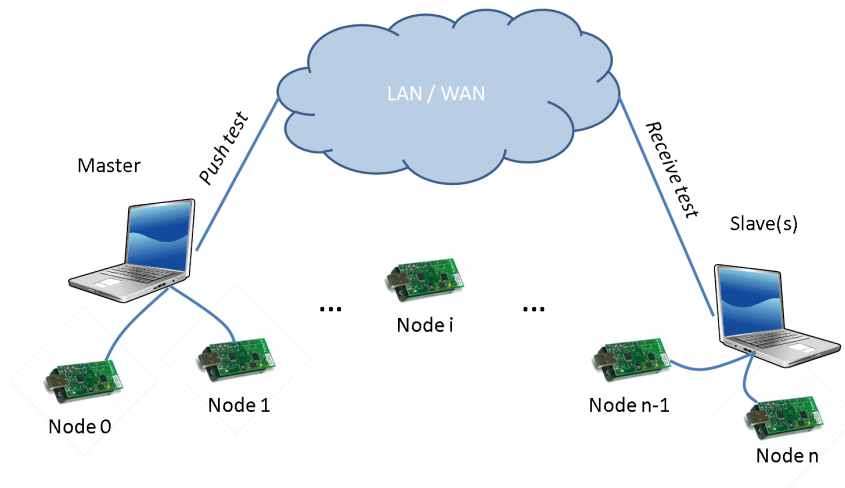


Figure 2: Testbed Topology

- i. generates a Makefile which expresses all the input parameters, which can be then understood by the compiler. For example, a transmission power must be translated to an integer value that is conform to the radio chip datasheet.
- ii. compiles the test program into a binary adapted to the sensor platform CPU architecture.
- iii. disseminates the test binary image to each node in the testbed, with the help of the slave computers if necessary.
- iv. runs the program test until completion.
- v. receives the results of the test via the serial port, which is attached to the sensor mote, and updates the table sheet with these results.

The controller is implemented in Java and communicates to the slave nodes via web services.

3. Data Analysis

- (a) Once all the data is collected during the measurement phase, the table sheet is imported into Matlab. There we proceed to statistical analysis by finding relations between the input and performance parameters, which is further explained in Section 5.5.

5.3 Individual tests

Unfortunately, it is often not possible to measure all the desired performance parameters in one test. Therefore, we split the measurements phase into several tests, each test being responsible for a subset of the desired performance variables. In the following sections, we discuss some of these tests.

5.3.1 Latency test

In the latency test, the nodes in the network form a chain. The first node is connected to the master controller, and initiate the test by sending a packet to the next node in the chain, and starts a timer at the same time. Each node forwards the packet along the chain. When it reaches the end of it, the packet is sent backwards, hop-by-hop to the first node, which upon reception stops the timer. The time measured is the performance variable we want to measure with this, and is commonly referred as the round-trip time (RTT) metric. The test is specially relevant if the MAC layer relies on LPL techniques, where the duty cycle can impact drastically the RTT value.

For the results in this milestone, we only used two nodes for the test, as the delivery ratio was poor. Adding more nodes increases exponentially the end-to-end losses, which makes the test impracticable.

5.3.2 Throughput test

The throughput test measures how the network throughput is affected in burst converge-cast transmission. In this scenario, the network forms a star, with the measuring node in the middle. All leave nodes transmit packets

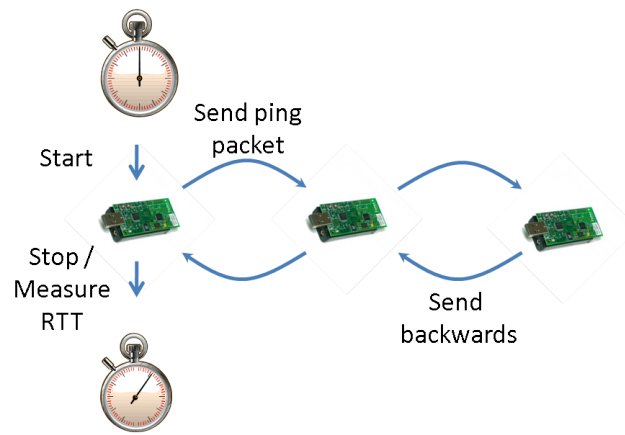


Figure 3: Latency Test

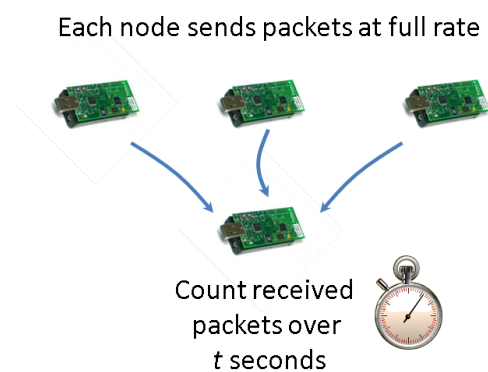


Figure 4: Throughput Test

at full speed to the root, which counts the number of receiving packets per second over a period of time. This test permits before all to see how much collision resistant is the MAC layer.

In this milestone, we used three transmitting nodes all located within transmission range of each other.

5.4 Analysis Methodology

After the experiments are conducted the resulting test data is provided in an Excel sheet containing both parameter values and performance metrics. The data is imported into Matlab and analyzed there.

First a pure statistical assessment is performed where correlations are assessed with the help of time-series plots, scatter plots and evaluation of correlation coefficients. Examples of these plots and values are given in section 5.5. Using this simplistic assessment, it can be determined if the experiments had a positive outcome and more advanced analysis methods can be applied.

While the plots yield provide a qualitative result from visual inspection by the user, the correlation coefficient provide a more quantitative measure on the significance of the relationship between a parameter and a performance metric.

Thereafter, a factor analysis and process modeling is conducted. These methods need to be adjusted during the initial setup and automated for larger setups with a more complex network topology. When process models are available, these models are assessed using relative gain arrays [6], interaction index arrays [5] and coherence functions [8].

5.5 Results

The test data from the first tests are analyzed using the above described methodology. To the date of writing the first statistical screening of the data has been conducted.

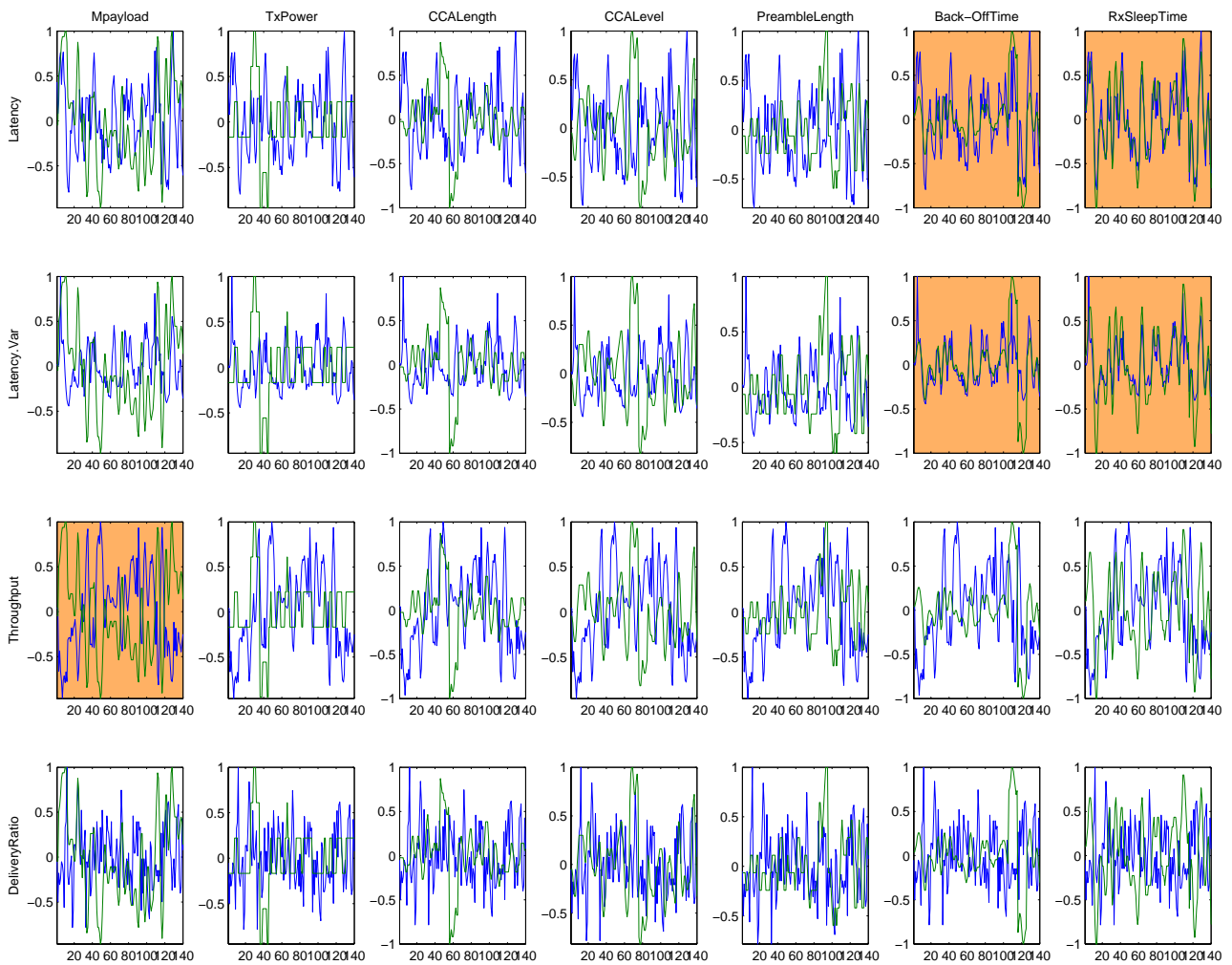


Figure 5: Time series plot of the min-max scaled data sets. Rows related to the performance metrics and columns to the parameters.

In Figure 5 the time series plot for the experiment is given. If a parameter and performance metrics relate largely with each other, then the parameter line (green) and the performance line (blue) will have a good overlap and will follow each other well. From visual inspection it can be seen that the shaded frames to have a good resemblance.

In Figure 6 the scatter plot for the experiment is given. If a parameter and a performance metrics have a linear relationship, then the data points will be collected close to line with a certain slope. From visual inspection it can be seen that the shaded frames form line-like shapes.

For the data sets the correlation coefficients are calculated. These coefficients represent a normalized value of the zeroth lag correlation between to performance metrics and a parameter and are given in Table 3. Again it can be seen that the framed numbers are large (close to one) which yields are large correlation.

It can be concluded that the initial experiments are successful and that correlations between parameters and

Table 3: Correlation coefficients

Parameter	MPayload	TxPower	CCALength	CCALevel	Preamble Length	Back-Off Time	RxSleepTime
Latency	0.1923	-0.1037	-0.0653	-0.2619	-0.0543	0.6879	0.8883
Latency.Var	0.0536	-0.1227	-0.0707	-0.3445	-0.1419	0.6872	0.8937
Throughput	-0.9549	0.0486	0.2136	-0.1523	-0.0056	-0.0289	-0.0155
Delivery Ratio	0.0677	0.0823	-0.0576	0.1262	0.1317	-0.4105	-0.4484

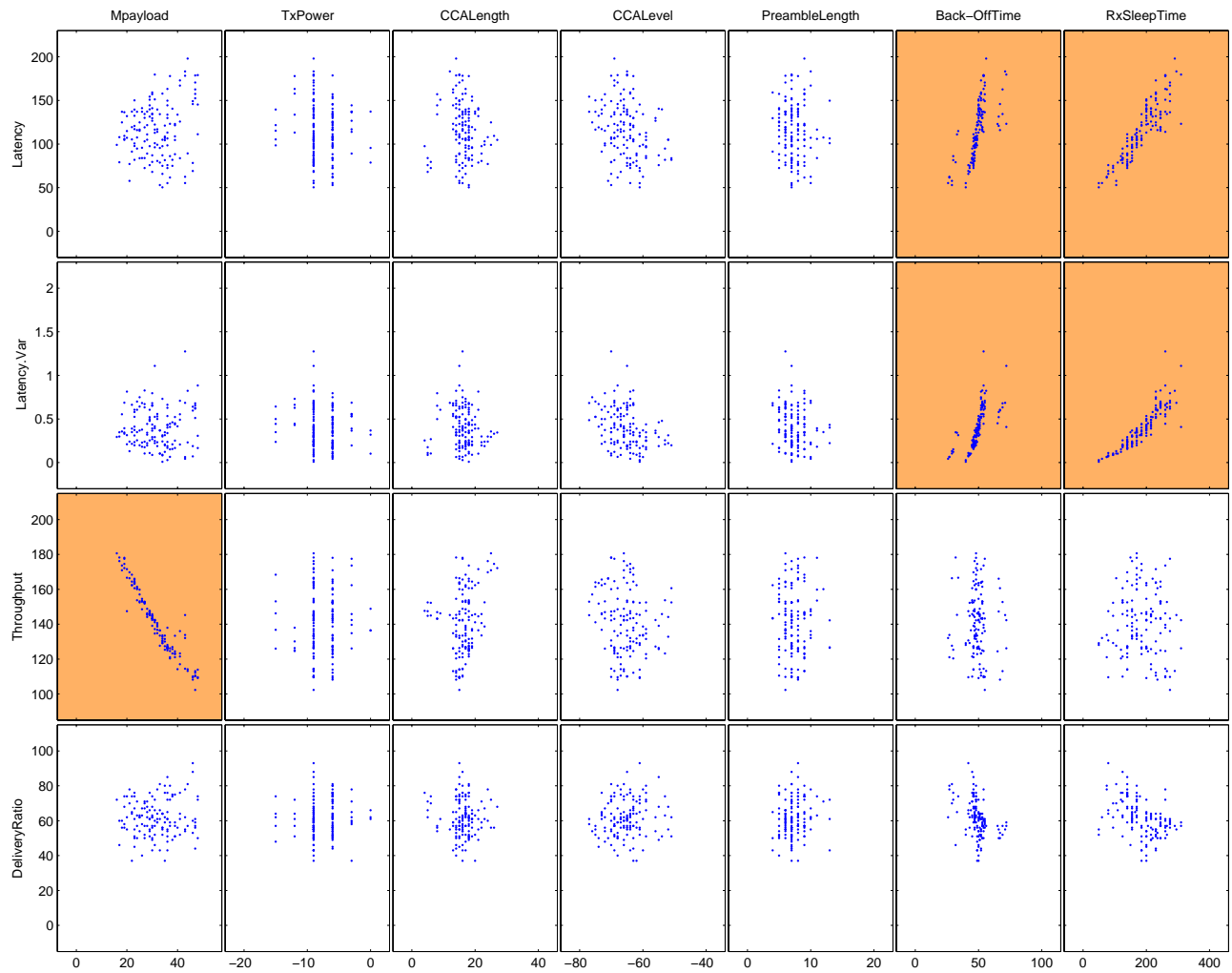


Figure 6: Scatter plot of the data sets. Rows related to the performance metrics and columns to the parameters.

performance metrics can be determined directly from the experimental data. It can be seen that Back-Off Time and RxSleepTime have a large effect on Latency and its variance, and that the MPayload has a large effect on the throughput.

It is important to note that this first analysis is only assessing one-on-one relationships and does not consider any combination effects which are usually available in MIMO processes. Therefore, the more advanced analysis methods have to be applied to get an understanding of the exact significance of parameters on performance metrics.

6 Modularization of MAC layer

In the previous section we analyzed the impact of different adjustable parameters of a CSMA-based MAC protocol on latency, throughput and delivery ratio system-level performance metrics. These parameters are present in virtually all MAC protocols. The existing MAC protocols differ from each other by specific functional components which include these parameters. In order to be able to assess suitability of one or another MAC scheme to a particular application the components should be described in a unified way.

In the nutshell we consider network protocols as *controllers* (in automatic control terms) of a communication process. These controllers are composed from a set of generic components with a number of parameters each. The functionality of a protocol is observed through its control actions, which affect the communication process and lead to a modification of its performance characteristics. A specific controller is synthesized from a number of interconnected generic components, which constitutes the structure of the controller. The choice of the component parameters can be understood as the tuning of the controller. Selection and interconnection of components, and the choice of the parameters result determine the behavior of the protocol and will be denoted *control law*.

In this section we present a methodology for component based modeling of network protocols on the example of Medium Access Control protocols. We show how different instances of MAC protocols (known as monolithic solutions) could be achieved only by parameterizing abstract MAC controllers. We demonstrate the correctness of our model by matching the control laws for controller instances derived using our components to the results of analytic performance analysis of two monolithic protocols: X-MAC [7] and B-MAC [41] previously published in the literature.

6.1 Components of Medium Access Control protocol

In the context of this document, *component* is a system element offering a pre-defined functionality, and interacts with other components. The major property of a component is its *reusability*. Therefore, a component implements common functionality on some level of granularity, which can be reused by other components. Besides the description of its functionality each component is characterized by a set of properties and expose adjustable parameters. Further on we differentiate two types of the components:

- **Basic components** implement certain control function, not further decomposed into subcomponents. Furthermore in the article basic component notation $BC_{_}$ precedes the name of the component.
- **Logical component** implement the logic of interactions between other logical and basic components. Further in the article logical component notation $LC_{_}$ precedes the name of the component.

In order to derive a model for generic MAC service we analyzed existing traditional MAC protocols in the Internet [26] as well as the MAC schemes designed specifically for wireless sensor networks ([15, 17] and references therein). We first extracted the major common functional patterns. We then described these patterns as parameterized components and identified control laws when applicable. After that we semantically classified the components and formally described relationships between them using ontologies. Combining the control laws of the components included in the protocol we could derive a law for the control function. This section presents a general structure of the MAC controller, the identified components and their classification. We describe examples of modeling abstract MAC services in the next section.

6.2 From functional patterns to components

The analysis of the existing MAC protocols shows that all of them consist of two phases: A channel establishment and the data exchange phases. What essentially makes the different protocols different is the algorithmic and data structure contents of these phases. For example, pure CSMA/CA schemes the data transmission is preceded by clear channel assessment and preamble (or RTS/CTS) exchange while in generic TDMA or FDMA based MAC, slot (or frequency) schedules should be exchanged before exchanging data. These schedules in some hybrid protocols [24] are in turn exchanged over a CSMA channel management scheme.

In the message exchange phase some protocols do not acknowledge the reception of data while other do. Acknowledgement defines the reliability of the given protocol. The reliability can be further categorized in explicit or implicit. There are also protocols that combine several channel establishment schemes and as a result have several types of message exchange procedures[24, 47].

A component based view of a generic MAC protocol on the top level of abstraction is shown graphically in Fig. 7(a). In the figure each logical component is steered by its specific *LC_LifePattern* component. This component actually implements the logic of the corresponding component by alternating execution of other logical and basic components in time. In the figure the *LC_LifePattern* component of *LC_MAC* component alternate execution of the channel access implemented by one or several *LC_ChannelManager* components and the actual data transmission implemented by one or several *LC_MsgExch* components. In turn, the *LC_LifePattern* component of a channel manager, implementing a reliable, 1-persistent, CSMA based channel access, alternates execution of basic components responsible for periodic sending of channel request control messages and receiving the channel established messages¹. Fig.7(b) illustrates an example of identifying logical components in X-MAC protocol.

6.3 Classification and formal representation of components

In order to enable functionally consistent composition and formal verification of component based protocols the components should be semantically classified and the relationships between components indicated and formally characterized.

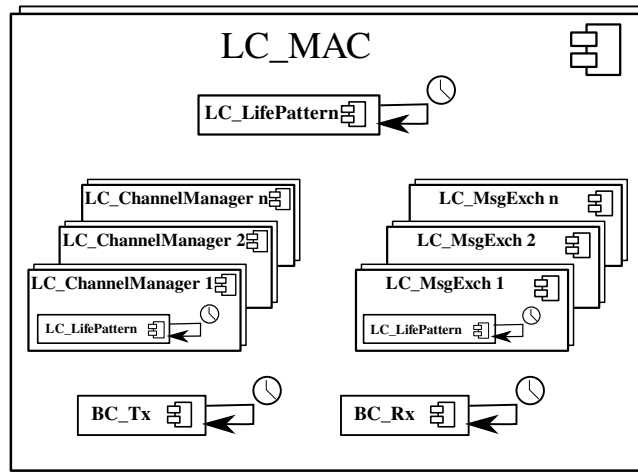
In this article we do not argue in favor of one or another methodology for data representation and reasoning. As a illustration of a technique that can be deployed in our framework we choose tree based ontologies to systematically represent the set of functional components.

Fig. 8 presents a simplified ontology of components needed for modeling of MAC protocols. For further usage we encoded the ontology trees as shown by numbers in curly brackets. Further on we address a particular component by its code in the ontology tree. For example, $0.0(MsgExchType, PersistencyType)$ denotes a CSMA/CA component with two parameters. We describe parametrization in the following section. The dotted line indicate the relationships between different components eligible to be combined into the particular instance of the MAC protocol. Table 4 describes the functionality, parameters and properties of selected components that we use in the showcase example in the next section.

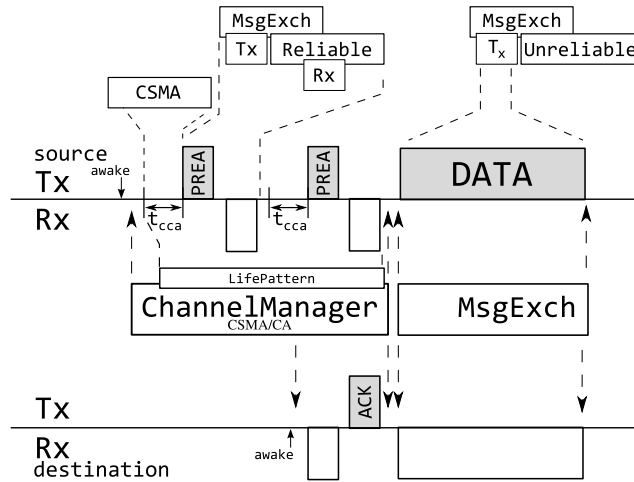
6.4 Analytical validation of component based modeling

The goal of this section is to illustrate the logic behind modeling of MAC protocols through various parametrizations of components on different levels of abstraction and validate the correctness of the composition. As an example we model two abstract instances of CSMA/CA protocol from the transmitter perspective. The first instance consists of a reliable channel manager and unreliable data exchange. The second instance has a mirrored functionality: an unreliable channel manager and reliable data exchange. We validate the correctness of the composition by deriving the control laws of the two controllers in the time domain (T_{TX}) and comparing the result to the previously published time properties of the known protocols.

¹In different MAC protocols channel request and confirmation messages may be denoted differently (e.g. *preamble* in X-MAC and *RTS* in S-MAC stand for channel request messages and *ACK* in X-MAC and *CTS* in S-MAC stand for channel established messages) while bearing exactly the same semantics.



(a) MAC logical component on highest level of abstraction.



(b) Identification of functional patterns in X-MAC.

Figure 7: Component based MAC.

Composing MAC instance 1

For the reliable channel manager we select logical component 1.0.0(L_{msg}, L_{ACK}) that sends a sequence of bits of length L_{msg} to request a channel and expects reception of an acknowledgement of length L_{ACK} . The send function is implemented by directing the output of the transmit component 2.1.0(t_{wait}, L_{msg}) to the transceiver controller 2.2.0(R). The t_{wait} parameter in component 2.1.0 equals the sum of the duration of the contention window and the time needed for the clear channel assessment procedure. The receive function is implemented by directing the output of the transceiver controller 2.2.0(R) to the "wait-and-receive" component 2.1.1(t_{wait}, L). In this case t_{wait} is the duration of the waiting timer. The functionality of the reliable message exchange component is completed by the "life pattern" controller 2.0.0(t_p, k) which implements 1-persistent channel access method with periodic retransmission. Finally, our controller in the message exchange phase transmits the data message unreliably by directing the output of the transmit component 2.1.0(t_{wait}, L) to the transceiver controller 2.2.0(R), where L is the length of the data packet. Fig.9 illustrates the intermediate steps when deriving the control law for the first instance of the CSMA/CA protocol.

Composing MAC instance 2

By mirroring the used components inside the channel manager and the message exchange components we design control law for the second instance of the CSMA/CA controller. Which have the same exact form as previously. The only difference is in its parameterization. $t_{wait}^{2.1.0}$ and $t_{wait}^{2.1.1}$ denote CSMA access time and acknowledgement waiting time correspondingly. This time, however, $L_{msg}^{1.0.0}$ is the length of the data packet to be transmitted reliably and $L_{msg}^{1.1}$ is the length of the channel request control message.

Table 4: Description of selected components, their parameters and properties.

Component	Description	Properties
0.0(<i>MsgExchType</i> , <i>PersistencyType</i>)	Establish a communication channel between two nodes using CSMA/CA Method. Implement by performing exchange procedure of control messages of type <i>MsgExchType</i> according to persistency model <i>PersistencyType</i> .	Logical component.
1.1(L_{msg})	Generate a message of size L_{msg} bits, submit to TX component.	Logical component.
1.0.0(L_{msg} , L_{ACK})	Generate a message of size L_{msg} bits, submit to TX component and wait for reception of implicit acknowledgement of size L_{ACK} .	Logical component.
2.0.0(t_p , k)	This component executes another component up to k times periodically with period t_p . k is an adjustable parameter, t_p is derived from the time properties of the executed components. This component contributes to the value of <i>Back - OffTime</i> parameter in Table 3.	Basic component. Control function in time domain is $k \times t_p$.
2.0.1(t_u , k)	Similar to 2.0.0 but the time interval between two executions increases exponentially starting with initial time unit t_u . k is an adjustable parameter, t_u is derived from the time properties of the executed components. This component contributes to the value of <i>Back - OffTime</i> parameter in Table 3.	Basic component. Control function in time domain is $2^k \times t_u$.
2.1.0(t_{wait} , L)	Waits time t_{wait} and submit a data structure of size L to the transceiver component. This component contributes to the value of <i>MPayload</i> parameter in Table 3.	Basic component. Control function in time domain is $t_{wait} + T_{2.2.0}(L)$. t_{wait} parameter, depending on the type of the channel access method, may include the time needed for clear channel assessment and the time for switching transceiver between different states.
2.1.1(t_{wait} , L)	Waits time up to time t_{wait} until receiving interrupt occurs and terminate. Continues to receive the message if interrupt occurs at the edge of wait timer expiration. This component contributes to the value of <i>RxSleepTime</i> parameter in Table 3.	Basic component. Control function in time domain is $\begin{cases} t_{wait}, if(1) \\ t_{wait} + T_{2.2.0}(L), if(2). \end{cases}$ Where (1) is condition of either failed reception or successful reception happen within t_{wait} and (2) is condition of receiving a receive interrupt at the edge of the wait timer expiration.
2.2.0(R)	A transceiver component performing actual transmission and reception of a bit sequence with rate R . Parameter R is adjustable by choosing appropriate hardware element.	Basic component. Control function in time domain is $[bitsequence]/R$.

Analytical validation of the controller composition

In fact, the presented above controllers implement the functionality of X-MAC and B-MAC respectively. This is easy to verify by first appropriately setting the adjustable parameters as shown in Table 5. The resulting functions match the ideal delay analysis in [48, 29].

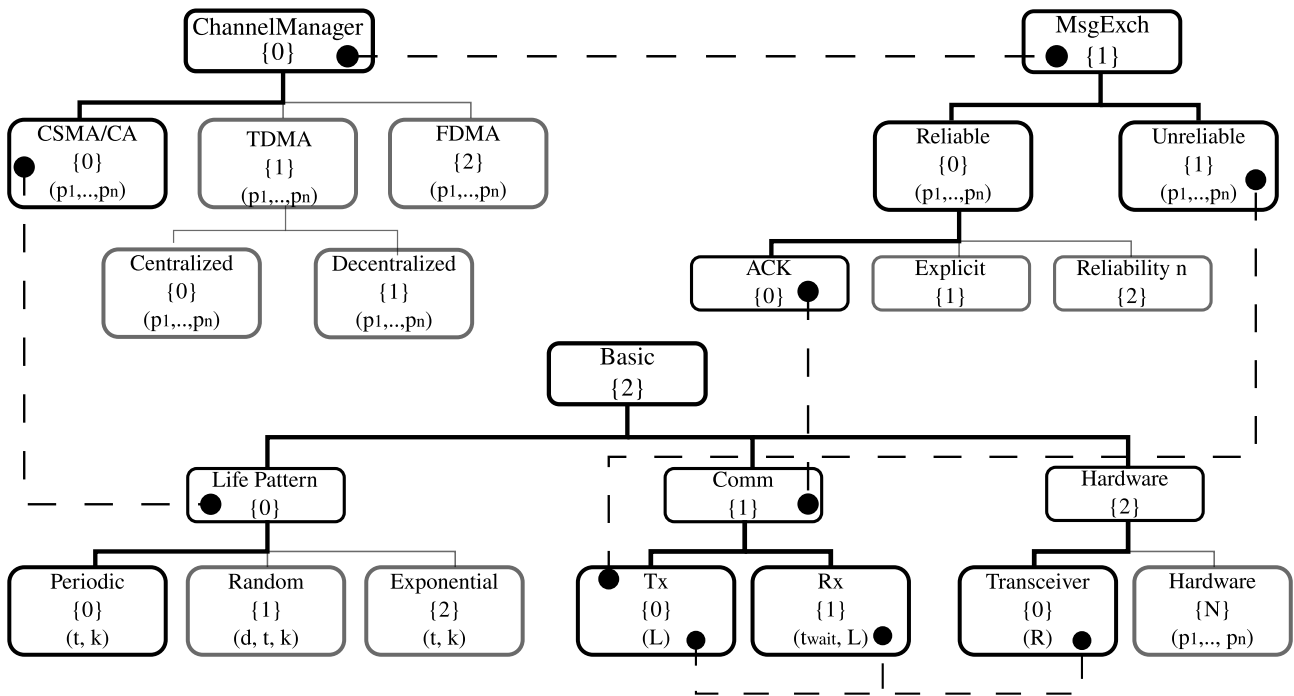


Figure 8: Simplified tree ontologies for MAC components. By bold lines the nodes needed for presenting an example of composition are shown.

Table 5: Parameter description for XMAC and BMAC in our model.

Param.	XMAC	BMAC
$L_{msg}^{1.0.0}$	length of strobe preamble	length data message
k	# of times to send preamble	1
$L_{msg}^{1.1}$	message size	length of the preamble
L_{ACK}	size of acknowledgement	size of acknowledgement
$t_{wait}^{2.1.0}$	contention window and clear channel assessment	contention window and clear channel assessment
$t_{wait}^{2.1.1}$	retransmission timer	retransmission timer

Thus, we have shown the two instances of MAC protocols known as monolithic solutions can in fact be represented as one controller by applying different parameterization.

7 Discussion and future work

On controller design problem formulation

We formulated a problem of choosing the right structure for the network controller as selecting an optimal combination of components and setting the correct values of adjustable parameters which jointly lead to desired performance characteristics of the communication process. On the example of CB-modeling of only one (and in fact the simplest) class of MAC protocols the number of adjustable parameters is more than seven. When modeling more complex MACs and moreover when modeling combination of protocols on different communication layers the number of adjustable parameters will grow dramatically. Not all adjustable parameters, however, equally contribute to the change in control variables. Therefore the problem of identifying the most significant parameters should be solved prior to the main optimization process. As a matter of fact there is still a lack of methodology for the selection of both the structure and the parameters.

Assuming that the relationship between the adjustable parameters and the controlled variables can be described by a parametric model, it is possible to apply the concept of interaction measures to analyze the signifi-

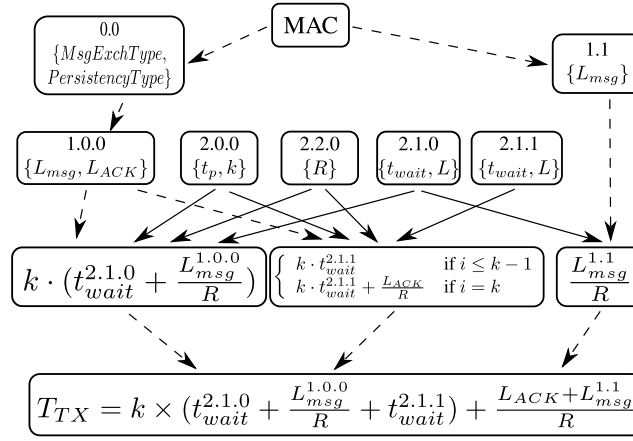


Figure 9: Derivation of the control law T_{TX} for MAC instance 1. The left branch indicated by dashed arrows shows compositions of the channel manager and the right branch of composition of the message exchange component.

cance of parameters on the performance metrics. Recent developments within the area of multivariable control suggest index arrays for the quantification of significance [8, 5]. These index arrays make use of the propagation of effect. As a result it can be determined which variables have the most significant effect on a performance indicator and thus, should be used for tuning of the performance.

On parametric MIMO model of communication process

Clearly, a communication process with many peers is a multiple-input and multiple output process, which largely depends on complex interactions in the communication channel. By describing the process in mathematical terms, these interactions can be characterized and assessed.

Using this assessment it should be possible to determine structures of a protocol and also optimize setting of the adjustable parameters in a more systematic way.

Currently, we are applying the presented concept on simplistic network topologies, principally on a link between two nodes with no interference from other nodes. The next step would be to increase the complexity of considered networks by introducing disturbances in the communication environment. In order to be applicable in a real-life installation site, it is stipulated that the models for the environment need to be derived in order to assess the effect of the disturbances.

On CB-modeling and implementation of network protocols

The first challenge associated with this part of the dependability framework is the large variety of network protocols of different complexity. We so far classified simple MAC protocols. Classification of all protocols is both intellectually intensive and time consuming.

The second challenge concerns adopting the presented CB model for direct implementation. When matter comes to implementing several distinct MAC schemes implementing the protocols with reusable components would allow code-efficient combination of several techniques in bridge nodes. It is important that the correct and the consistency of the CB implementation should be formally verified [19, 32, 22].

8 Summary of Deliverable 1.3

In this document we presented a systematic approach towards designing modularized protocols and ranking a contribution of their components to the overall system performance. In the nutshell, this approach is based on three steps: 1.) identifying adjustable parameters in existing protocols, 2.) ranking their influence on the system-level performance metrics and 3.) defining protocol modules exposing the parameters of the highest rank.

To this end we presented the details of definition and ranking the components of MAC protocols. We also overviewed the ranking methods of functional blocks for protocols on the routing layer. It is important to note, that the task of modularization is complex and requires substantially more efforts beyond those planned for this work package. The work described in this document, therefore, will be continued in work package WP3 along the lines of identifying new functional components and further ranking of the impact of the their adjustable parameters on the overall system performance.

References

- [1] Ahmed A. Ahmed, Hongchi Shi, and Yi Shang. A survey on network protocols for wireless sensor networks. In *Proc of the International Conference on Information Technology: Research and Education, 2003.*, 2003.
- [2] G. Almes, S. Kalidindi, and M. Zekauskas. A One-way Delay Metric for IPPM. RFC 2679, 1999.
- [3] G. Almes, S. Kalidindi, and M. Zekauskas. A One-way packet loss for IPPM. RFC 2680, 1999.
- [4] G. Almes, S. Kalidindi, and M. Zekauskas. A round trip Delay Metric for IPPM. RFC 2681, 1999.
- [5] Wolfgang Birk and Alexander Medvedev. A note on gramian-based interaction measures. In *Proc. of European Control Conference ECC, Cambridge, UK*, pages 2660–5, 2003.
- [6] E.H. Bristol. On a new measure of interaction for multivariable process control. *IEEE Transactions on Automatic Control*, 11:133–134, January 1966.
- [7] Michael Buettner, Gary V. Yee, Eric Anderson, and Richard Han. X-mac: a short preamble mac protocol for duty-cycled wireless sensor networks. In *SenSys '06: Proceedings of the 4th international conference on Embedded networked sensor systems*, pages 307–320, New York, NY, USA, 2006. ACM.
- [8] Miguel Castaño and Wolfgang Birk. New methods for structural and functional analysis of complex processes. In *IEEE Multiconference on Systems and Control, in St. Petersburg*, 2009.
- [9] Anna Chaltseva and Evgeny Osipov. Empirical cross-layer model of TCP throughput in multihop wireless chain, 2009.
- [10] L. Chen, S. H. Low, and J. Doyle. Joint congestion control and media access control design for ad hoc wireless networks. In *Proc of IEEE Infocom, Miami, USA, 2005*, 2005.
- [11] Mung Chiang. Balancing transport and physical layers in wireless multihop networks: Jointly optimal congestion control and power control. *IEEE Journal on Selected Areas in Communications*, 23(1), 2005.
- [12] Mung Chiang, Steven H. Low, A. Robert Calderbank, and John C. Doyle. Layering as optimization decomposition: A mathematical theory for network architectures. *Proceeding of the IEEE*, 95(1):255–312, January 2007.
- [13] David D. Clark. Modularity and efficiency in protocol implementation. RFC 817, 1982.
- [14] Tyson Condie, Joseph M. Hellerstein, Petros Maniatis, Sean Rhea, and Timothy Roscoe. Finally, a use for componentized transport protocols. In *HotNetsIV: Fourth Workshop on Hot Topics in Networks*. ACM, 2005.
- [15] P.P. Czapski. A survey: Mac protocols for applications of wireless sensor networks. *TENCON 2006. 2006 IEEE Region 10 Conference*, pages 1–4, Nov. 2006.
- [16] C. Demichelis and P. Chimento. IP Packet Delay Variation Metric for IP Performance Metrics IPPM. RFC 3393, November 2002.
- [17] I. Demirkol, C. Ersoy, and F. Alagoz. Mac protocols for wireless sensor networks: a survey. *Communications Magazine, IEEE*, 44(4):115–121, April 2006.
- [18] C. Doerr, M. Neufeld, J. Fifield, T. Weingart, D.C. Sicker, and D. Grunwald. Multimac - an adaptive mac framework for dynamic radio networking. In *New Frontiers in Dynamic Spectrum Access Networks, 2005. DySPAN 2005. 2005 First IEEE International Symposium on*, pages 548–555, Nov. 2005.
- [19] Mohamed Eltoweissy and Mohamed Younis. Dependable wireless sensor networks. *Computer Communications*, 29(2):149–150, 2006. Dependable Wireless Sensor Networks.

- [20] A. Farago, A.D. Myers, V.R. Syrotiuk, and G.V. Zaruba. Meta-mac protocols: automatic combination of mac protocols to optimize performance for unknown conditions. *Selected Areas in Communications, IEEE Journal on*, 18(9):1670–1681, Sep 2000.
- [21] David Gay, Philip Levis, Robert von Behren, Matt Welsh, Eric Brewer, and David Culler. The nesc language: A holistic approach to networked embedded systems. In *Proceedings of the ACM SIGPLAN 2003 conference on Programming language design and implementation*, pages 1–11. ACM Press, 2003.
- [22] Michael G. Hinchey, James L. Rash, Christopher A. Rouff, and Denis Gracanin. Achieving dependability in sensor networks through automated requirements-based programming. *Computer Communications*, 29(2):246–256, 2006. Dependable Wireless Sensor Networks.
- [23] Norman C. Hutchinson and Larry L. Peterson. The x-kernel: An architecture for implementing network protocols. *IEEE Trans. Softw. Eng.*, 17(1):64–76, 1991.
- [24] Global Inventures/ZigBee. Zigbee alliance – home page. <http://www.zigbee.org>, November 2009.
- [25] B. Johansson and M. Johansson. Mathematical decomposition techniques for distributed cross-layer optimization of data networks. *IEEE Journal on Selected Areas in Communications*, 2006.
- [26] Holger Karl and Andreas Willig. *Protocols and Architectures for Wireless Sensor Networks*. Wiley-Interscience, 2007.
- [27] V. Kawadia and P.R. Kumar. A cautionary perspective on cross-layer design. *Wireless Communications, IEEE*, 12(1):3–11, Feb. 2005.
- [28] Kevin Klues, Gregory Hackmann, Octav Chipara, and Chenyang Lu. A component-based architecture for power-efficient media access control in wireless sensor networks. In *SenSys '07: Proceedings of the 5th international conference on Embedded networked sensor systems*, pages 59–72, New York, NY, USA, 2007. ACM.
- [29] Mikko Kohvakka. *Medium Access Control and Hardware Prototype Designs for Low-Energy Wireless Sensor Networks*. PhD thesis, Tampere University of Technology, P.O. Box 527, FI-33101, 2009.
- [30] Mikko Kohvakka, Mauri Kuorilehto, Marko Hännikäinen, and Timo D. Hämäläinen. Performance analysis of ieee 802.15.4 and zigbee for large-scale wireless sensor network applications. In *Proceedings of the 3rd ACM international workshop on Performance evaluation of wireless ad hoc, sensor and ubiquitous networks, Terromolinos, Spain.*, pages 48 – 57, 2006.
- [31] R. Koodli and R. Ravikanth. One-way Loss Pattern Sample Metrics. RFC 3357, August 2002.
- [32] Sandeep S. Kulkarni and Mahesh Arumugam. Transformations for write-all-with-collision model,. *Computer Communications*, 29(2):183–199, 2006. Dependable Wireless Sensor Networks.
- [33] J. Mahdavi and V. Paxson. IPPM Metrics for Measuring Connectivity. RFC 2678, 1999.
- [34] M. Mathis and M. Allman. A Framework for Defining Empirical Bulk Transfer Capacity Metrics. RFC 3148, July 2001.
- [35] E. Meshkova, J. Riihijarvi, J. Ansari, K. Rerkrai, and P. Mahonen. An extendible metadata specification for component-oriented networks with applications to wsn configuration and optimization. pages 1–6, Sept. 2008.
- [36] A. Morton, L. Ciavattone, G. Ramachandran, S. Shalunov, and J. Perser. Packet Reordering Metrics. RFC 4737, November 2006.
- [37] P. Naur and B. Randel. Software engineering: Report of a conference sponsored by the nato science committee. *Scientific Affairs Division*, page 231, 1969.

- [38] Evgeny Osipov. tinylunar: One-byte multihop communications through hybrid routing in wireless sensor networks. In *In New2AN 2007*, 2007.
- [39] Evgeny Osipov and Christian Tschudin. Ingress throttling and its applications in ieee 802.11 based manets. *Journal of communication software and systems*, 2, 2007.
- [40] C. Perkins, E. Belding-Royer, and S. Das. Ad hoc On-Demand Distance Vector (AODV) Routing. RFC 3561 (Experimental), July 2003.
- [41] Joseph Polastre, Jason Hill, and David Culler. Versatile low power media access for wireless sensor networks. In *SenSys '04: Proceedings of the 2nd international conference on Embedded networked sensor systems*, pages 95–107, New York, NY, USA, 2004. ACM.
- [42] Karl-Erik Årzén, Anton Cervin, Håkan Hjalmarsson, Krister Jacobsson, Björn Johansson, Karl Henrik Johansson, Mikael Johansson, Niels Möller, João Sousa, John Lygeros, A. Panousopoulou, G. Nikolakopoulos, A. Tzes, Luigi Palopoli, Antonio Danesi, Adriano Fagiolini, and Antonio Bicchi. D6.1 requirements report on control-aware embedded networks. Technical report, European Commission, FP6 IP Runes, IST-004536-RUNES, 2005.
- [43] Christian Tschudin. Flexible protocol stacks. In *SIGCOMM '91: Proceedings of the conference on Communications architecture & protocols*, pages 197–205, New York, NY, USA, 1991. ACM.
- [44] Haitao Wu, Yunxin Liu, Jie Chen, and Qian Zhang. Layer 2.5 softmac: end-system based media streaming support on home networks. In *Global Telecommunications Conference, 2005. GLOBECOM '05. IEEE*, volume 1, pages 5 pp.–, Nov.-2 Dec. 2005.
- [45] Haitao Wu, Yunxin Liu, Qian Zhang, and Zhi-Li Zhang. Softmac: Layer 2.5 collaborative mac for multimedia support in multihop wireless networks. *Mobile Computing, IEEE Transactions on*, 6(1):12–25, Jan. 2007.
- [46] Haitao Wu, Xin Wang, Yunxin Liu, Qian Zhang, and Zhi-Li Zhang. Softmac: layer 2.5 mac for voip support in multi-hop wireless networks. In *Sensor and Ad Hoc Communications and Networks, 2005. IEEE SECON 2005. 2005 Second Annual IEEE Communications Society Conference on*, pages 441–451, Sept., 2005.
- [47] Wei Ye, J. Heidemann, and D. Estrin. Medium access control with coordinated adaptive sleeping for wireless sensor networks. *Networking, IEEE/ACM Transactions on*, 12(3):493–506, June 2004.
- [48] Suyoung Yoon. *Power Management in Wireless Sensor Networks*. PhD thesis, North Carolina State University, USA, January 2007.
- [49] Yan Yu, Ramesh Govindan, and Deborah Estrin. Geographical and energy aware routing: A recursive data dissemination protocol for wireless sensor networks. Technical report, UCLA, 2001.
- [50] Sandro Zampiri. Trends in networked control systems. In *Proc of the 17th IFAC World Congress, Seoul, Korea, July 6-11, 2008*, pages 2886 – 2894. The International Federation of Automatic Control, July 2008.
- [end of document]