# Poster Abstract: Exploiting Protocol Models for Generating Feasible Communication Stack Configurations

Marco Zimmerling, Federico Ferrari, Matthias Woehrle, and Lothar Thiele
Computer Engineering and Networks Laboratory
ETH Zurich, Switzerland
{zimmerling,ferrari,woehrle,thiele}@tik.ee.ethz.ch

## ABSTRACT

Communication stacks are composed of distinct layers that, in principle, operate independently and interact through well-defined interfaces. However, resource constraints in sensor networks typically necessitate optimizations, leading to implicit assumptions and dependencies among layers (*e.g.*, a collection protocol assumes the MAC protocol provides sufficient bandwidth). These dependencies are often tracked manually, yet become extremely complex as protocols evolve and requirements change. We propose to model assumptions and dependencies explicitly, as constraints on protocol parameters. This allows for using standard tools to generate feasible protocol configurations. We demonstrate the effectiveness of our approach using the example of FTSP running on top of a low-power listening MAC protocol.

## Categories and Subject Descriptors

C.4 [**Performance of Systems**]: *Modeling techniques*; D.2.1 [**Software Engineering**]: Requirements/Specifications—*Methodologies*

## General Terms

Performance, Reliability

## Keywords

Protocol Configuration, Constraint Programming

## 1. INTRODUCTION

Keeping track of the various interactions and dependencies between the different layers of a communication stack is a critical yet inherently complex task. Choi *et al.* [1] tackle the problem of inter-protocol interference on the network layer and propose an isolation layer to avoid collisions and guarantee protocol fairness. Indeed, unintended interactions are even more likely between protocols on different layers: protocols frequently make assumptions on the behavior of other protocols to meet the application requirements [6].

We propose to explicitly model assumptions of protocols to expose hidden dependencies between individual layers. While creating such a model is a considerable initial overhead, it frees the designer from managing the various dependencies manually. Moreover, it allows for automatically generating feasible and consistent configurations of the complete stack for *i)* different protocol combinations, *ii)* different deployment settings, *iii)* different protocol parametrizations, and *iv)* when requirements change.


*IPSN'10,* April 12–16, 2010, Stockholm, Sweden.
ACM 978-1-60558-955-8/10/04.

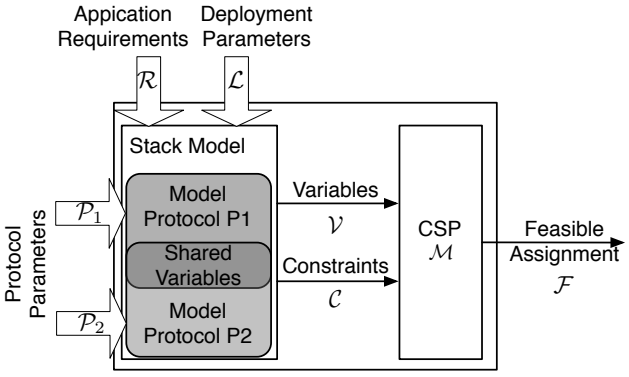


**Figure 1: Overview of the approach.**

## 2. APPROACH

Figure 1 outlines our approach. The stack model comprises different protocol models and their individual parameters $\mathcal{P}_i$ (*e.g.*, the sleep interval of the MAC protocol). Shared variables appearing in more than one protocol model signal dependencies and interactions among protocols. In addition, the stack model takes as inputs deployment parameters $\mathcal{L}$ (*e.g.*, the number of nodes) and a set of application requirements $\mathcal{R}$ (*e.g.*, the required data yield). From this model, we extract a set of constraints $\mathcal{C}$ on variables $\mathcal{V}$ for which we seek to find feasible assignments $\mathcal{F}$, the protocol configurations.

In particular, we can formulate a constraint satisfaction problem (CSP) $\mathcal{M} = \langle \mathcal{V}, \mathcal{D}, \mathcal{C} \rangle$, where $\mathcal{V}$ is an $n$-tuple of decision variables $\mathcal{V} = \langle v_1, v_2, \ldots, v_n \rangle$, $\mathcal{D}$ is a corresponding $n$-tuple of domains $\mathcal{D} = \langle D_1, D_2, \ldots, D_n \rangle$ such that $v_i \in D_i$, and $\mathcal{C}$ is a $k$-tuple of constraints $\mathcal{C} = \langle C_1, C_2, \ldots, C_k \rangle$ [3]. A constraint $C_i$ is a pair $\langle R_i, S_i \rangle$, where $S_i$ is a $k_i$-tuple of variables in $\mathcal{V}$ and $R_i$ is a relation of arity $k_i$ over the domains of the variables in $S_i$. Solving the CSP $\mathcal{M}$ involves finding a value for each variable in $\mathcal{V}$, where the constraints $\mathcal{C}$ specify that some subsets of their domains $\mathcal{D}$ cannot be used together. We can use standard CSP solvers for this task, which combine interference (constraint propagation) and search algorithms. If we are interested in an optimal protocol configuration, $\mathcal{M}$ can be transformed into a constraint optimization problem (COP) by adding an objective function.

Our approach is modular and allows for fast iterations when application requirements or deployment characteristics change. Since every protocol parameter can also be specified as a constraint, we can analyze the effect of deviating conditions. Most importantly, the stack model and thus the generated protocol configurations capture all protocol dependencies, which reduces the risk of unintended interactions at runtime.

| Set | Model input | Description | Value/Domain |
|---|---|---|---|
| $\mathcal{L}$ | $H$ | Max. hop distance to sink | 5 |
| | $\hat{\vartheta}$ | Max. clock drift variation | $10^{-8}$ s$^{-1}$ |
| $\mathcal{P}$ | $T_l$ | LPL MAC listening time | 4 ms |
| | $T_s$ | LPL MAC sleeping time | $[T_s^{\min}, T_s^{\max}]$ |
| | $T_f$ | FTSP synchroniz. period | $[T_f^{\min}, T_f^{\max}]$ |
| | $N$ | FTSP table size | 8 |
| $\mathcal{R}$ | $D \leq D^{\max}$ | Max. radio duty cycle | $D \leq 2\%$ |
| | $A \geq A^{\min}$ | Min. accuracy | $A \geq (40 \text{ ppm})^{-1}$ |

**Table 1: Model inputs of FTSP/LPL MAC feasibility study.**

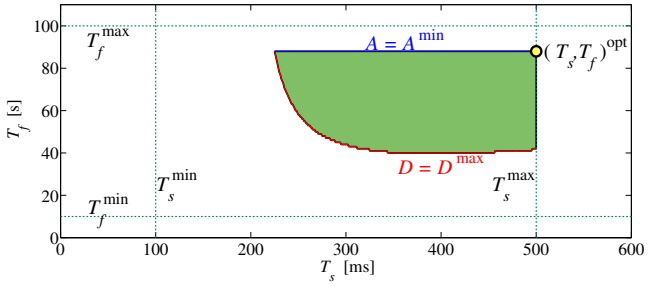


**Figure 2: Generated feasible region and optimal configuration.**

# 3. FEASIBILITY STUDY

We demonstrate our approach using the example of FTSP [4] running on top of a low-power listening (LPL) MAC protocol.

**Protocol Background.** FTSP is a time synchronization protocol for sensor networks. It synchronizes the local clocks of the nodes to a common time reference by periodically exchanging time-stamped messages and performing linear regression on the timestamps received in the last $N$ periods. The synchronization period $T_f$ determines how often synchronization messages are exchanged. A shorter $T_f$ results in a higher synchronization accuracy but also in an increased communication. While FTSP regularly triggers the exchange of synchronization messages, the MAC protocol is responsible for actually broadcasting these messages.

Using a LPL MAC protocol, nodes sleep most of the time but wake up regularly for a short period to poll the channel. Nodes stay awake if they detect channel activity to handle an incoming transmission; otherwise, they go back to sleep. The sender of a broadcast keeps on transmitting the data packet for a period slightly exceeding the sleep interval so that all its neighbors definitely wake up once to receive the broadcast. The length of the sleep interval $T_s$ determines how often nodes poll the channel. Therefore, it also determines for how long a sender must transmit the broadcast packet.

**Dependencies.** Running FTSP on top of a LPL MAC protocol, we find a fundamental trade-off between synchronization accuracy and energy consumption. While nodes should synchronize frequently (short $T_f$) to maximize accuracy, they should sleep as much as possible (long $T_s$, where $T_f \gg T_s$) to save energy. Parameters $T_f$ and $T_s$ must be set prior to deployment to implement the best trade-off for the application at hand. Our approach assists by generating a feasible configuration; that is, it assigns values from given, valid domains to $T_f$ and $T_s$ such that *i)* the application requirements on energy and synchronization accuracy and *ii)* the implicit bandwidth requirements of FTSP are satisfied.

**Stack Model and CSP Formulation.** In [2] we derive an analytical stack model of energy consumption and synchronization accuracy of FTSP running on top of a LPL MAC protocol.

$$D = \frac{T_s}{T_f} + \left(1 - \frac{T_s}{T_f}\right) \frac{T_l}{T_l + T_s} \quad (1)$$

$$A = [H\hat{\vartheta}T_f(N+1)]^{-1} \quad (2)$$

We use the fraction of time $D$ the radio is on during a synchronization period $T_f$ as a measure of the MAC's energy efficiency, which is given by (1). The accuracy of the drift estimation $A$ corresponds to the synchronization accuracy of FTSP, as given by (2). Both protocol models share one variable: the synchronization period $T_f$. This clearly indicates a dependency of FTSP on the underlying MAC, namely that the radio is turned on often enough to exchange synchronization messages. Moreover, the stack model projects the energy-accuracy trade-off on variables $T_s$ and $T_f$.

We formulate a CSP using decision variables $\mathcal{V} = \langle T_f, T_s \rangle$ and domains $\mathcal{D} = \langle [T_f^{\min}, T_f^{\max}], [T_s^{\min}, T_s^{\max}] \rangle$. All other variables in (1) and (2) are constant with the values listed in Table 1. The set of constraints $\mathcal{C}$ corresponds to the application requirements $\mathcal{R}$ on energy consumption $D$ and synchronization accuracy $A$, also shown in Table 1. We handle the bandwidth requirements of FTSP by a proper definition of the domains: $T_s \leq T_s^{\max} \ll T_f^{\min}$.

**Implementation.** We model and solve the provided CSP using the ECL$^i$PS$^e$ [5] constraint programming system. ECL$^i$PS$^e$ is based on Prolog and comes with a rich set of built-in libraries, which allows for specifying the complete constraint program in less than 20 lines of code. To speed up the solving process, we discretize the domains of $T_s$ and $T_f$.

We let ECL$^i$PS$^e$ generate solutions to the CSP, which results in the feasible region shown in Figure 2. Picking a protocol configuration $(T_s, T_f)$ from this region ensures that the application requirements as well as the implicit bandwidth requirements of FTSP are satisfied. Furthermore, curves $D = D^{\max}$ and $A = A^{\min}$ expose the non-trivial effects of parameters $T_s$ and $T_f$ on radio duty cycle and synchronization accuracy.

By adding (1) as the objective function to minimize, given the accuracy constraint $A \geq A^{\min}$, ECL$^i$PS$^e$ finds the optimal protocol configuration $(T_s, T_f)^{\text{opt}} = (0.5\text{ s}, 88.8\text{ s})$. This results in a minimum radio duty cycle of $D^{\text{opt}} \approx 1.36\%$ that still provides the desired synchronization accuracy of $A^{\min} = (40 \text{ ppm})^{-1}$ on drift estimation.

**Acknowledgments.** The work presented here was supported by the National Competence Center in Research on Mobile Information and Communication Systems (NCCR-MICS), a center supported by the Swiss National Science Foundation under grant number 5005-67322.

# 4. REFERENCES

[1] J. Il Choi, M. A. Kazandjieva, M. Jain, and P. Levis. The case for a network protocol isolation layer. In *Proceedings of SenSys'09*, pages 267–280, 2009.

[2] F. Ferrari, M. Zimmerling, and L. Thiele. Accuracy and duty-cycle of FTSP on a LPL-MAC. Technical Report 319, ETH Zurich, 2010.

[3] E. C. Freuder and A. K. Mackworth. Constraint satisfaction: An emerging paradigm. In *Handbook of Constraint Programming*, pages 13–27. 2006.

[4] M. Maroti, B. Kusy, G. Simon, and A. Ledeczi. The flooding time synchronization protocol. In *Proceedings of SenSys'04*, pages 39–49, 2004.

[5] M. Wallace, S. Novello, and J. Schimpf. ECLiPSe: A platform for constraint logic programming, 1997.

[6] Q. Zhang and Y.-Q. Zhang. Cross-layer design for QoS support in multihop wireless networks. *Proceedings of the IEEE*, 96(1):64–76, 2008.