Uncertainty Aware Hybrid Clock Synchronisation in Wireless Sensor Networks

Christoph Steup, Sebastian Zug, Jörg Kaiser
Department of Distributed Systems
Faculty of Computer Science
Otto-von-Guericke-University
Magdeburg, Germany
Email: {steup,zug,kaiser}@ivs.cs.ovgu.de

Andy Breuhan
Rohde & Schwarz DVS GmbH
Email: andybreuhan@andybreuhan.de

***Abstract*—Wireless Sensor Networks, aiming to monitor the real world's phenomena reliably, need to combine and post-process the detected individual events. This is not possible without reliable information of the context of the individual event. One such information of very high importance is time. It enables ordering of events as well as deduction of further data like rates and durations. An unreliable time base influences not only the ordering of events, but also the deduced values, which in consequence are unreliable as well. Therefore the synchronization of the clocks of the individual nodes is of high importance to the reliability of the system. On the other hand tight and reliable synchronization typically induces a large message overhead, which is often not tolerable in WSN scenarios. This paper proposes a new hybrid synchronization mechanism enabling tight synchronization in single hop environments and looser synchronization in multi hop environments. The lack of a guaranteed synchronization precision is mitigated by an explicit synchronization uncertainty, which is passed to the application. This enables the application to react to changes in the current synchronization precision.**

Keywords*—Wireless Sensor Networks, Time Synchronization, Uncertainty

I. Introduction

Wireless Sensor Networks (WSN) gain increasing attention by researchers as well as industry and governments. They provide the ability to monitor large areas for events efficiently and with small effort. Two examples are the SafeCast project [1], which aims to provide people with the ability to cheaply monitor radiation in their vicinity and share this data with others, and the project aiming to detect forest fires, endangering nature and people, as described by Yu et al. [2].

Even though WSN are often reduced to disseminating data, evaluation and decision making are equally important for these systems. Catastrophe warning systems for example, need robust decision making mechanisms to be accepted by people. Therefore, reliable post processing and context detection are crucial to provide a robust output. One of the most important context attributes needed for WSN is time, since an unreliable time base influence the ordering as well as the deduction of events, which in consequence become unreliable as well. Consequently, a reliable, precise global time base is a must-have for each WSN detecting safety relevant events. However, the granularity of the time base depends on dynamics of the system as well as the requirements of the application.

Typical time synchronization approaches like the Network Time Protocol (NTP) [3] or the Precision Time Protocol (PTP) [4] require a lot of messages and consider the underlying network to be quite robust, which are properties not available in typical WSN. This led to the creation of specially adopted time synchronization protocols. Typically these try to provide a trade-off between message overhead and synchronization precisions. Additionally they try to tolerate message losses and changes in the topology of the network. However, most of the existing protocols either try to provide a tight synchronization in a single hop environment and degrade heavily in multi hop environments or provide a generally looser synchronization in both. Unfortunately, none of the existing protocols provide the application with information on the current status of the synchronization, which might be degraded by errors in communication or heavy changes in topology.

This paper introduces a new hybrid synchronization mechanism for WSN. It provides tight synchronization in single hop environments and looser synchronization with a decreasing precision based on the topological distance between nodes. It enables applications to adapt to the currently achieved synchronization precision by providing an estimated synchronization uncertainty together with every time stamp.

The description of our approach starts with a discussion of related work in Section II, followed by the description of our concept in Section III. In Section IV we describe our implementation within the Omnet++ network simulator, the tests carried out and their results. The paper closes with a conclusion of our results and some ideas on future work in Section V.

II. State of the Art

In order to assess the current state of clock synchronisation for WSN, we describe 6 approaches representing basic concepts in the following section.

A. Reference Broadcast Synchronization (RBS)

Reference Broadcast Synchronization as described by Elson et al. [5] is a synchronization mechanism exploiting a physical broadcast in a shared medium. The synchronization starts with one node transmitting a *NOW*-message to all other nodes. This message serves as an indication for all nodes to take a local time stamp. Afterwards the timestamps are exchanged between all nodes. Finally, all nodes compute individually their offset towards the mean of all exchanged timestamps.

This mechanism reduces the critical path to the transmission time of the *NOW*-message and the local processing time on each node until the local time stamp is taken. This provides very tight synchronization in single hop scenarios as long as the computation time is bounded. However, in worst case for n nodes $\mathcal{O}(n^2)$ messages are needed for a single synchronization round.

Additionally, RBS reacts very sensitive to the mobility of nodes. This is caused by the used averaging mechanism of the protocol. The contribution of all nodes to the averaged new time can create large shifts in the local clock of each node whenever one node's clock is far off. This is especially problematic if this node is only a temporary member of the broadcast group.

B. Delay Measurement Time Synchronization Protocol (DMTS)

The Delay Measurement Time Synchronization Protocol described by Ping [6] extends RBS by exploiting low-level hardware access. It extends the *NOW*-message with a time stamp taken and inserted just before sending. Therefore the exchange of the individual local time stamps can be omitted and the message count can be heavily reduced. To reach similar synchronization precision as RBS, the author estimates the delay of the transmission analytically and modifies the inserted time stamp accordingly.

This approach solves the large amount of message necessary for a single synchronization of RBS. However, in-depth knowledge of the needed hardware and communication mechanisms as well as low-level hardware access is needed to use it. Additionally, the problem of faulty or mobile nodes of RBS is enforced since only a single time stamp is communicated, which hinders fault recovery mechanism to be established.

C. Continuous Clock Synchronization in Wireless Real-time Applications (CCS)

The Continuous Clock Synchronization for Wireless Real-time Applications by Mock et al. [7] is a Master-Slave synchronization method extending the basic clock synchronization mechanism of the 802.11 standard [8]. In contrast to the basic mechanism, the clocks of the slaves are not simply set to the time stamp of the master, but are gradually adopted by adjusting their rate. Additionally, the precision of the synchronization is enhanced by dividing the time beacon in a *NOW*-message and an additional message containing the time stamp of the *NOW*-message. This division enables a more exact estimation of the master's time finishing the transmission of the *NOW*-message. The additional message needed can be saved if the master's time stamp is incorporated in the next *NOW*-message.

This approach enables continuous clock synchronization without gaps in the time base. Additionally, it provides better precision than the basic 802.11 synchronization mechanism without additional message overhead.

D. Probabilistic Clock Synchronization Service (PCS)

The Probabilistic Clock Synchronization Service by PalChaudhuri et al. [9] is an extension of RBS enabling a dynamic trade-off between synchronization precision and message overhead. This approach transmits n *NOW*-messages in one synchronization round, which are used to derive the skew of the sender's and the receiver's clock though linear regression. The results are combined and broadcasted back to the receivers in range. By comparing their own data with the data received from the sender, they are able to adopt their own clocks. To derive the number of needed *NOW*-messages the authors assumed the synchronization error to be normal distributed with zero-mean and a standard deviation of σ. Based on this distribution, the authors analytically derive the probability $P(|\epsilon| < \epsilon_{max})$ of the synchronization error to be less than a specified value ϵ_{max}. For a specified probability of the synchronization to be more precise than ϵ_{max}, the authors derive the number n of message needed. This number heavily depends on the standard deviation σ of the normal distribution.

The approach provides a dynamic trade-off between message overhead and synchronization precision. However the trade-off depends on the standard deviation of the synchronization error, but the acquisition of this value was not covered by the authors. Additionally, only a mathematically prove without any simulation was conducted to evaluate the idea. Consequently, the authors never discussed the effects of non-normal distributed synchronization errors.

E. Time Synchronization in Ad-Hoc Networks (TSAN)

Römer's Time Synchronization in Ad-Hoc Networks [10] is based on Christian's Algorithm [11]. It estimates the round trip time of a message between sender and receiver. Whereas Christian's Algorithm proposed a dedicated server for clients to communicate to, Römer attaches time stamps to events communicated in the network. Therefore Römer's algorithm ideally induces a zero message overhead. However, not all events are acknowledged by the receiver, which might create large durations between events flowing in both directions between two nodes. This is mitigated by the insertion of additional dummy events in case the duration grows too large.

TSAN applies a very loose multi hop synchronization with an ideal message overhead of 0. Unfortunately, the real message overhead is heavily dependent on the actual communication in the network and is therefore very hard to estimate for a real system.

F. Event Composition in Time-Dependant Systems (ECTS)

Liebig et al. [12] described a way to combine multiple events even though their time stamps might not be exact. To achieve this they extended a time stamp to a time interval and derived an ordering relation $<$ for time intervals. Together with a known uncertainty of the event's time stamp, ordering might be possible even in loosely synchronized systems. However, this transition changed the order relation to a half order relation. Consequently, there might be situations in which two events cannot be ordered. This is the case if the intervals of the events' time stamps overlap.

This approach, even though it is not directly handling clock synchronization, enables uncertainty aware clock synchronization algorithms to provide time intervals, which provide awareness of synchronization precision to higher layer applications.

G. Summary

The individual problems and features of the protocols are summarized in Table I. As visible none of the described approaches fully solve the problem of multi-hop uncertainty-aware clock synchronization in wireless sensor networks. However, each approach contains individual features, which might enhance the performance of our approach.

Our approach incorporates the beneficial properties of the different approaches in a single clock synchronization mechanism, which is uncertainty and topology aware and produces time intervals usable by an application. The time interval algebra exploit Liebig et al.'s ordering relation, see II-F, to enable an seamless integration in existing technology. The next section describes our mechanism in detail.

TABLE I. COMPARISION OF THE DISCUSSED TIME SYNCHRONIZATION PROTOCOLS.

Protocol	Synchronization precision	Multihop capability	Message Overhead	Robustness
RBS	high	none	$\mathcal{O}(n^2)$	fragile
DMTS	high	medium	$\mathcal{O}(n)$	fragile
CCS	high	medium	$\mathcal{O}(1)$	robust
PCS	medium	good	dynamic	medium
TSAN	low	good	0 - $\mathcal{O}(1)$	medium

III. UNCERTAINTY AWARE CLOCK SYNCHRONISATION (UACS)

For an efficient synchronization of clocks in WSN multiple parameters are important. On one hand, the synchronization needs to be scalable, while on the other hand the overhead may not exceed a certain threshold to safe battery and prevent an overload of the network. Most of the approaches discussed in Section II favour one over the other. However if we limit our self to certain base topologies better solutions might be found. One interesting topology is the cluster tree structure of IEEE 802.15.4 networks [13] in beacon-enabled mode. This mode divides the nodes in groups called Personal Area Networks (PANs), which have an individual coordinating instance managing the internal communication. The individual PANs communicate only through their respective coordinators, as visible in Figure 1. In the remaining section of the paper we consider an 802.15.4 network, with an already established cluster tree structure. The formation and the handling of dynamic changes in this structure are not considered in this paper.

Based on the initial assumption, that clock synchronization may have a decreasing precision based on topological distance between nodes in the network, we propose a hybrid clock-synchronization, consisting of a tight synchronization mechanism for each individual PAN called Intra Cluster Synchronization and a loose synchronization mechanism between the individual PAN Coordinators, called Inter Cluster Synchronization.

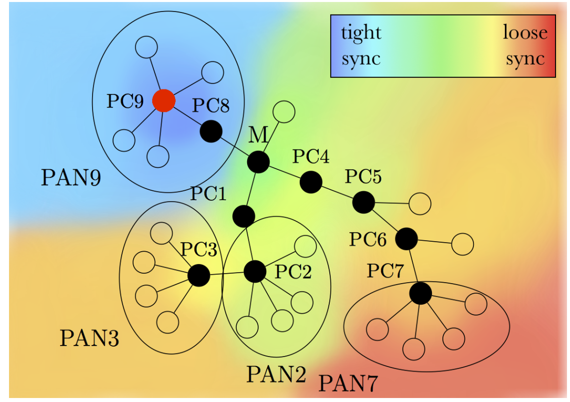

Fig. 1. Example cluster tree structure of an IEEE 802.15.4 Network. The colours indicate the topological distance between each node and $PC9$.

A. Intra Cluster Synchronization

The Intra Cluster Synchronization is based on the CCS approach, see Section II-C. Therefore, each PAN slave $P_{s_j} \in Slaves$ has a virtual synchronized clock $VC_{s_j}(t)$. This clock uses the time stamps created by the node's hardware clock $C_{s_j}(t)$ and modifies it based on the current rate $\rho_{s_j,i}$:

$$VC_{s_j}(t) = \rho_{s_j,i}\left(C(t)_{s_j} - C(t_i)_{s_j}\right) + VC_{s_j}(t_i) \quad (1)$$

The task of the Intra Cluster Synchronization is the estimation of the parameter $\rho_{s_j,i}$ for each slave at each synchronization round i. The 802.15.4 standard allows a PAN Coordinator to attach additional information to the beacon frame. We use this to attach a 64bit time stamp $t_{c,i}$ to each beacon b_{i+1} transmitted by the coordinator. The attached time stamp represents the coordinators time of successful transmission of the last beacon. This time stamp together with the local reception time of the last beacon $t_{s_j,i}$ is then evaluated by each slave P_{s_j} to compute a new rate $\rho_{s_j,i}$.

As described by DMTS, see Section II-B, hardware knowledge may be used to provide the needed local time stamps. The 802.15.4 standard provides the `PD-Data.confirm` primitive as a local event indicating completion of a transmission. The time of this event is used as the source of the time stamp $t_{c,i}$. On reception of the beacon each PAN slave P_{s_j} takes a local time stamp $t_{s_{j,i+1}}$. The networks tightness τ together with the internal computation time t_{comp} of the nodes limits the accuracy of the local time stamps. This computation time can also be mitigated by the PD-DATA.indication primitive of the 802.15.4 standard. Therefore we omit t_{comp} in our approach and the time difference of creation of the local time stamps is bounded by τ.

After acquiring the time stamp for the actual synchronization round the PAN slaves compute the offset $o_{j,i} = t_{c,i} - t_{s_j,i}$ between their previous local time stamp $t_{s_j,i}$ and the time stamp transmitted through the beacon $t_{c,i}$. This is used to compute a new rate $\rho_{s_j,i-1} = k_\rho o_{j,i}$ for the node's virtual clock to compensate the offset, with k_ρ being a proportional factor controlling the rate of adoption.

The Intra Cluster Synchronization provides continuous clock synchronization between the PAN Coordinator and its slaves. The overhead is minimal since no additional message

 ISBN: 978-1-61208-353-7

is necessary and the beacons are only slightly enlarged. The robustness of the synchronization mainly depends on the used algorithm to detect a crash and reselect a PAN Coordinator. The synchronization accuracy depends on the clock skew between the PAN Coordinator and its slaves as well as the tightness of the network and the beacon interval of the PAN. Mobility of individual slaves is handled by the adoption rate of the virtual clock and has no influence on other slaves of the same PAN.

B. Inter Cluster Synchronization

Diverging from the Intra Cluster Synchronization, see Section III-A, each PAN Coordinator does not modify its own virtual clock. Instead every event received by a PAN Coordinator P_{c_r}, which is transmitted by another adjacent PAN Coordinator P_{c_s} is transformed in the time domain of the PAN Coordinators virtual clock $VC_r(t)$, as proposed by TSAN, see Section II-E. To achieve this, the PAN Coordinator P_{c_r} needs to calculate a virtual clock $VC_{r,s}(t)$ for each adjacent PAN Coordinator P_{c_s}.

The virtual clocks are handled similarly to the Intra Cluster Synchronization, since all beacons of all adjacent PAN Coordinators P_{c_s} are received by PAN Coordinator P_{c_r}. On reception of beacons, containing a time stamps $t_{s,i}$, P_{c_r} acquires a local time stamp $t_{r,s,i+1}$. This enables the computation of the offset $o_{r,s,i} = t_{s,i} - t_{r,s,i}$ between P_{c_s} and P_{c_r}. Afterwards P_{c_r} updates the rate $\rho_{r,s,i} = k_\rho o_{r,s,i}$ for the virtual clock $VC_{r,s}(t)$ towards P_{c_s}.

On reception of an event e_n from P_{c_s} containing a time stamps $e_n.ts_s$, P_{c_r} is able to transform the time stamp of the event to its own virtual clock $VC_r(t)$. The transformation is done by adding the offset between the Virtual Clock of the sender $VC_s(t)$ and the receiver $VC_r(t)$ to the event's time stamp:

$$e_n.ts_r = e_n.ts_s + VC_r(t) - VC_s(t) \tag{2}$$

This approach handles mobility well, since the mobility of a node in the local neighbourhood of P_{c_1} does not change P_{c_1}'s virtual clocks of the other nodes. Therefore the transformation of the events is independent of each other. However, the disadvantage is the accumulation of synchronization errors over multiple hops. Therefore we explicitly specify the synchronization error in a time interval $ti = [ts \pm \alpha], ts \in R^+, \alpha \in R^+$ replacing the time stamp $ts \in R^+$. Consequently, each hop additionally modifies the interval bounds by the currently estimated uncertainty of the synchronization $\alpha_{r,s}$:

$$e_n.\alpha_r = e_n.\alpha_s + \alpha_{r,s} \tag{3}$$

C. Estimating the Uncertainty

The estimation of the current uncertainty of the synchronization of the virtual clocks is difficult. Multiple factors influence the actual uncertainty in the synchronization, like beacon losses and the current drift of the individual clocks. In our approach the synchronization error $\epsilon_{r,s,i}$ of synchronization round i between two adjacent PAN Coordinators P_{c_s} and P_{c_r} is characterized by their offset $o_{r,s,i+1}$ at beginning of synchronization round $i + 1$.

Following PCS, see Section II-D, we model the synchronisation error to be a zero-mean Gaussian distribution $N(0, \delta)_{r,s}$. To estimate the standard deviation we use the synchronization errors of the previous n synchronization rounds as sample set $E_{r,s} = \{\epsilon_{i-n}, \epsilon_{i-n+1} \dots \epsilon_i\}$. We estimate the standard deviation $\delta_{r,s}$ of our zero-mean Gaussian based on the sample set. Based on this we compute the confidence interval $\left[\bar{x} \pm z_{\left(\frac{1+\gamma}{2}\right)} \frac{\delta}{\sqrt{n}}\right]$ of the synchronization with typical probability γ. The value $z_{\left(\frac{1+\gamma}{2}\right)}$ represents the $\frac{1+\gamma}{2}$-quantile of the standardised normal distribution. The resulting size of the confidence interval $\alpha_{r,s} = z_{\left(\frac{1+\gamma}{2}\right)} \frac{\delta}{\sqrt{n}}$ represents our current uncertainty estimation, which is added to current uncertainty of the event's time interval.

The complexity of this computation is only dependant on n, which represents a trade-off between estimation accuracy and memory and computation overhead. The quantile of the standardised normal distribution is a pre-defined constant, characterising the accuracy of the estimation.

D. Compatibility between Time Intervals and Time Stamps

Since time intervals and time stamps are not directly compatible we introduced a compatibility algebra $([ts \pm \alpha], \{+, -, \cdot, <\})$ based on interval arithmetic and the proposed half order relation of ECTS, see Section II-F. To additionally handle the deduction of new events, we added the operation to multiply the time interval with a constant $k \cdot [ts \pm \alpha] = [k \cdot ts \pm k \cdot \alpha]$. This is useful for applications to scale the difference between two time stamps as needed e.g. in the computation of speeds.

The resulting algebra establishes an ordered vector space, which is easy to compute even for deeply embedded systems. Transformations back to time stamps are easily possible by omitting the uncertainty part of the time intervals.

IV. Evaluation

We evaluated our uncertainty aware hybrid clock synchronization system with a simulation in the Omnet++ network simulator [14] version 4.2. Additionally we used the INETMANET network model [15] as well as the MiXiM model [16]. The implementation is distributed over two layers of the ISO/OSI stack. One part is located at layer 5 of the ISO/OSI stack and handles the transformation of time stamps for the Inter Cluster synchronization. The other is situated at layer 2 to gather high precision time stamps. Both layers are connected through a cross layer communication.

Our evaluation focuses on the Inter Cluster Synchronization, since single-hop synchronization is very well researched and our approach resembles CCS, see Section II-C, and DMTS, see Section II-B. Therefore the performance should be equivalent. For the Inter Cluster Synchronisation we evaluate two main topics. The first considers the influence of the beacon period on the precision of the synchronization. This test will provide information on the trade-off between message overhead and synchronization quality. The second test investigates the influence of the communication topology on the reachable multi-hop precision. It will evaluate the usability of the provided time stamps for smaller and longer routes. All tests used the internal 64 bit simtime of Omnet++ as reference for the synchronized clocks to evaluate the synchronization error. The simtime was modified by a randomly initialized drift($< 10^{-5}$), to provide a realistic clock for each node. The test considered 1000 randomly created routes between nodes in the network, which were created by an optimal routing algorithm.

Our simulation environment considers beacon losses, created by the collision of transmitted beacons of adjacent coordinators, and the resulting lack of information for the time synchronization. However, we did not transmit data events in the simulation. This decouples our simulations from the used MAC Algorithm and its parameters. Consequently, the current simulations consider an optimal MAC-Algorithm preventing all collision between beacons and events in the network.

A. Beacon Interval Analysis

The beacon interval analysis considered a rectangular grid of 50 PAN Coordinators. The area in which the nodes were distributed was $5000m$ times $5000m$. We used the 2.4GHz specification of the 802.15.4 standard at channel 11 with a maximum transmission power of $1mW$. The thermal noise was fixed at $110dBm$ and the receiver's sensitivity was set to $-85dBm$. Our simulation sweep started with a BO parameter of 8 up till the maximum allowed value of 14. The resulting beacon interval can be computed by $BI = \frac{16 \cdot 60 S \cdot 2^{BO}}{SymbolRate}$. The $SymbolRate$ of the 2.4GHz band of $65.2 \cdot 10^3 \frac{S}{s}$ results in beacon intervals between $3.8s$ and $241.2s$ length.

Figure 2 shows a Box-Whisker plot of the simulation's results. Results for BO values from 8 to 10 are omitted because of similarity. The boxes represent the bounds, where 50% of all values are included. The lines represent the interval containing 75% of all values and remaining data points are included as points. As visible with linear increasing BO values the mean synchronization error increases exponentially. This is to be expected because th beacon interval also increases exponentially. Additionally one observes a large standard deviation independent of the hop-count. This is caused by the unsynchronized beacons of the individual PAN Coordinators, which might collide and therefore increase the real beacon interval. Furthermore the data base is better for smaller hop-counts, since in the given scenario short routes are much more probable then longer routes.

This test proved the expected direct correlation between the beacon interval and the synchronization precision. Therefore this value is to be considered critical for the performance of the system.

B. Topology Analysis

Our second evaluation considers the performance of the system in different topologies. This is interesting, because topologies might have an influence on the length of the routes as well as the collision probability of the beacon frames. Therefore we considered four basic topologies with 200 nodes each. We choose a randomly generated, a grid, a linear and a circular topology. For this scenario we used the same parameters as for the Beacon Interval Analysis, see Section IV-A. This time the BO parameter was statically set to 8.

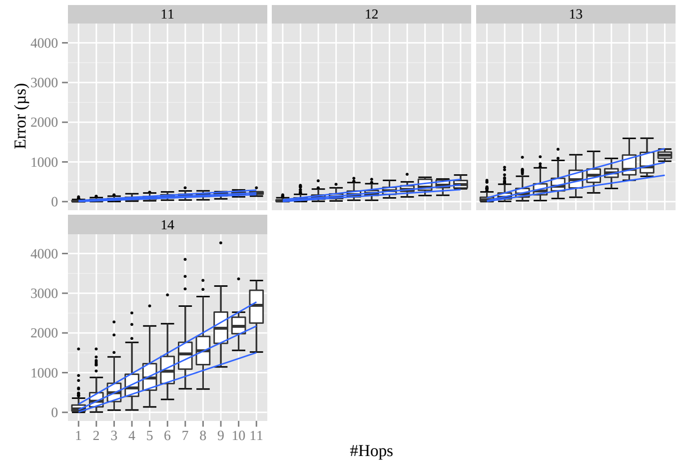

Fig. 2. Box-Whisker plot of the precision of a 50 node grid network with varying BO values.

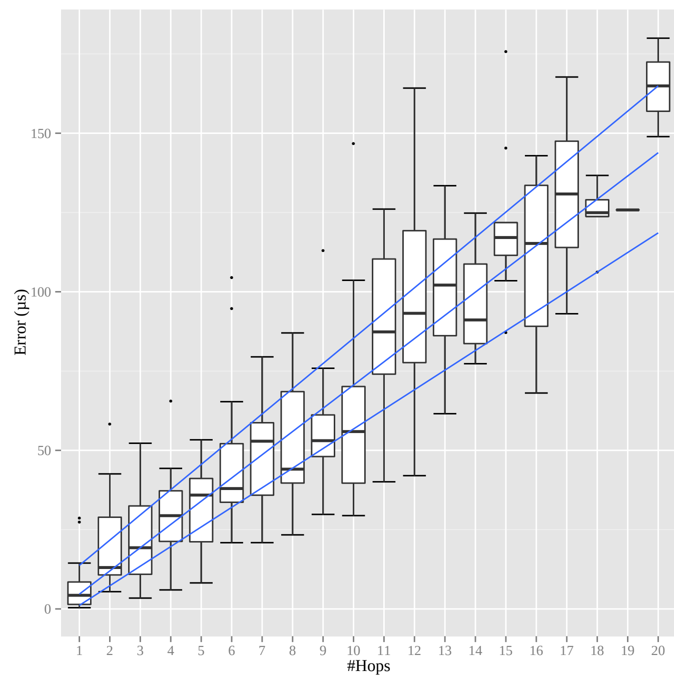

Fig. 3. Box-Wisker plot of the synchronization precision of a 200 Node random topology.

In Figure 3, the performance of the random topology is visible. The mean error of the simulation increases linear with the hop count. This is to be expected, since the synchronization error in the vicinity of each PAN is statistically the same. The summation of the uncertainties matches very well with the increasing error in the simulation. However the deviation of individual results is quite large in this case, which is caused by individual collisions of beacons. This problem is very dependent on the local setup of nodes around a PAN. Therefore it will increase the standard deviation of the random topologies synchronization error.

Table II shows an overview of the results of our evaluation of the different topologies. This table shows quite similar

 ISBN: 978-1-61208-353-7

TABLE II. SYNCHRONIZATION ERROR OF THE SIMULATION OF DIFFERENT TOPOLOGIES IN μs.

#Hop Count	Topology	Mean	Standard Deviation
1	Random	7.069008	0.008495
1	Grid	5.067219	0.005454
1	Linear	12.181786	0.009492
1	Circle	3.327730	0.003993
6	Random	47.401797	0.023945
6	Grid	26.574773	0.013489
6	Linear	55.861373	0.031209
6	Circle	21.579193	0.008327
11	Random	89.191069	0.027941
11	Grid	57.463772	0.019879
11	Linear	91.408097	0.023836
11	Circle	45.693382	0.015802
16	Random	110.255113	0.029990
16	Grid	80.882388	0.020756
16	Linear	131.582874	0.031220
16	Circle	73.628941	0.021878

results for the standard deviation of the tests for equally long routes in the different topologies. However, the mean of the synchronization error is quite different. As expected in all simulations the linear topologies have the largest mean error, which is cause by the highest collision probability of the beacons. The random topology performed the second worst, which is caused by local hot spots in the topology with a lot of nodes increasing the probability of beacon losses. In consequence all topologies show a linearly increasing error. Therefore our original assumption of an additive uncertainty fits very well to the simulated experiments.

C. Comparison with related protocols

Since the environments of the different described protocols differ, we compare our approach to protocols, with available multi-hop synchronization data. DMTS provided a mean synchronization error of $32\mu s$ for one hop and $46\mu s$ for two hop communication. Our approach performed better, but DMTS was evaluated on real hardware with a limited oscillator speed and computing power. Therefore, the results are not directly comparable. TSAN showed a mean synchronization error of $200\mu s$ for one hop and $1113\mu s$ for six hop communication. However, Römer et al. considered an unstructured network, whereas we exploited the structure of the network to increase the synchronization precision without message overhead.

V. CONCLUSION

This paper presents a novel hybrid clock synchronization approach that provides tight synchronization for local clusters of nodes as well as looser synchronization in multi-hop scenarios. The message overhead is minimal since existing periodic beacon messages of the 802.15.4 beacon-enabled mode were used to transmit the synchronization data. To handle the different synchronization precisions, uncertainty awareness was added to enable applications to decide in the case of ambiguous situations. The evaluation was done using the well-established network simulator Omnet++ in multiple scenarios with different configurations and supports the theoretical concepts of the described approach.

In future work, we want to evaluate the clock synchronization in a real scenario with real sensor nodes to evaluate the influence of unforeseen interference, as well as the limited processing power of the nodes. Additionally, we want to investigate the effect of the used MAC-Algorithm on the synchronization quality.

REFERENCES

[1] Y. Abe, "Safecast or the production of collective intelligence on radiation risks after 3.11," in *The Asia-Pacific Journal*, vol. 12, Issue 7, No. 5, Feb. 2014, pp. 1–6.

[2] L. Yu, N. Wang, and X. Meng, "Real-time forest fire detection with wireless sensor networks," in *Proceedings of the International Conference on Wireless Communications, Networking and Mobile Computing*, vol. 2, Sep. 2005, pp. 1214–1217.

[3] J. Burbank, D. Mills, and W. Kasch, "Network time protocol version 4: Protocol and algorithms specification," Internet Engineering Task Force (IETF), Tech. Rep. RFC 5905, Jun. 2010.

[4] "IEEE standard for a precision clock synchronization protocol for networked measurement and control systems," IEEE, Tech. Rep. Standard 1588-2002, 2002.

[5] J. Elson, L. Girod, and D. Estrin, "Fine-grained network time synchronization using reference broadcasts," in *SIGOPS*, vol. 36, no. SI. New York, NY, USA: ACM, Dec. 2002, pp. 147–163.

[6] S. Ping, "Delay measurement time synchronization for wireless sensor networks," Intel Research Berkeley Lab, Tech. Rep. IRB-TR-03-013, 2003.

[7] M. Mock, R. Frings, E. Nett, and S. Trikaliotis, "Continuous clock synchronization in wireless real-time applications," in *Proceedings of the 19th IEEE Symposium on Reliable Distributed Systems*, Oct. 2000, pp. 125–132.

[8] "IEEE Standard for Information Technology- Telecommunications and Information Exchange Between Systems-Local and Metropolitan Area Networks-Specific Requirements-Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," IEEE, Tech. Rep. IEEE Standard 802.11, 1997.

[9] S. PalChaudhuri, A. Saha, and D. B. Johnson, "Probabilistic clock synchronization service in sensor networks," in *IEEE Transactions on Networking*, vol. 2, no. 2, 2003, pp. 177–189.

[10] K. Römer, "Time synchronization in ad hoc networks," in *Proceedings of the 2nd ACM international symposium on Mobile ad hoc networking & computing*, ser. MobiHoc '01. New York, NY, USA: ACM, 2001, pp. 173–182.

[11] F. Cristian, "Probabilistic clock synchronization," in *Distributed Computing*, vol. 3, no. 3. Springer, Sep. 1989, pp. 146–158.

[12] C. Liebig, M. Cilia, and A. Buchmann, "Event composition in time-dependent distributed systems," in *Proceedings of the Fourth IECIS International Conference on Cooperative Information Systems*, ser. COOPIS '99. Washington, DC, USA: IEEE Computer Society, Sep. 1999, pp. 70–78.

[13] "IEEE Standard for Information Technology- Telecommunications and Information Exchange Between Systems- Local and Metropolitan Area Networks- Specific Requirements Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs)," Tech. Rep. IEEE Standard 802.15.4, 2006.

[14] G. Pongor, "OMNeT: objective modular network testbed," in *Proceedings of the International Workshop on Modeling, Analysis, and Simulation On Computer and Telecommunication Systems*. San Diego, CA, USA: Society for Computer Simulation International, Oct. 1993, p. 323–326.

[15] A. Ariza and A. Triviño, *Simulation of Multihop Wireless Networks in OMNeT++*. IGI Global, 2012, pp. 140–158.

[16] A. Köpke, M. Swigulski, K. Wessel, D. Willkomm, P. T. K. Haneveld, T. E. V. Parker, O. W. Visser, H. S. Lichte, and S. Valentin, "Simulating wireless and mobile networks in OMNeT++ the MiXiM vision," in *Proceedings of the 1st International Conference on Simulation Tools and Techniques for Communications, Networks and Systems & Workshops*. Brussels, Belgium: Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, Mar. 2008, p. 71:1–71:8.