Stochastic Battery Simulation

by

Eike Brandt

Florian Schaffarzyk

February 24, 2017

supervised by

Prof. Dr. Anna Förster

Dr.Ing. Asanga Udugama

Dipl.-Ing. Jens Dede

Contents

1 Motivation

Modelling batteries is a difficult, but nevertheless valuable task. Especially in mobile communication, where small devices with low energy consumption work for short time periods, the battery can recover energy within the idle phases and does thereby extent its lifetime.

Different models deal with this behaviour with different approaches. [1] gives an overview over some theoretical models that aim to simulate the recovery behaviour.

Within the course *Stochastic Simulation of Communication Networks and their Protocols* at the University of Bremen, hosted by the ComNets group of Prof. Anna Förster, several of this models have been implemented in the simulation program OMNeT++. This contribution deals with the *stochastic battery model* by Chiasserini and Rao, originaly published in [2–5].

Within this present study, first the theory of the stochastic model will be explained as it is developed by Chiasserini and Rao. In addition some minor tweaks will be presented that were made for the implementation in OMNeT++. Secondly, the realisation of the battery model will be explained as well as the framework within OMNeT++, which controls the energy consumption. Within this section also the most important parameters will be asserted and deduced how the final values were developed. In the end the model will be used to generate some analysis of different scenarios, as well as a discussion of the problems and limits of the model. It will be completed with a conclusion of the topic.

The OMNeT++ files can be found at

`https://github.com/brandte/stochastic_battery`

2 Theory of the battery model

Especially in small communication devices it is very important to maximize the battery life time. For sending and receiving data packets the device need a significant amount of energy whereas in sleep phase the device needs nearly none. Within a communication system these three actions of the device alternate to each other, which leads to an pulsed-discharge of the device.
In idle phases the battery can regain some parts of the previously drained energy. As result, the pulsed discharge, which can be found i.e. in wireless sensor networks makes it possible for the battery to recover energy within idle periods. The recovery process is based on electrochemical reactions that will not be covered here. For an overview look at [1], for a deeper study [6] is recommended.
To simulate this process different models considerate the *recovery effect*. In the following the stochastic model from Chiasserini and Rao [2–5] will be presented.

Stochastic model by Chiasserini and Rao

Chiasserini and Rao start in their first publications [2, 3] with the idea that the effects within a battery can be considered as stochastic. In this context, the capacity is modelled as discrete Markov-Chain with $N+1$ states. State 0 represents an empty battery whereas the battery with N charge units left marks a full battery. By draining the battery the discharge of the battery is shown in Figure 1 by the shrinking number of states. On the other side the recharging process is shown by going up the Markov-Chain.
The recovery process is thereby a stochastic process. To model the recovery as stochastic process the number of state N must be linked to the smallest amount of energy the device can drain from the battery. It must be given, that every energy usage can be displayed within the Markov-Chain.
In Figure 1 the first displayed method of the Markov-Chain is presented. Switching between the states is only possible by going one states up or down. It can be also seen, that each arrow is marked with the probability p,r or q, respectively. This shows that every step in the Markov-Chain is a stochastic process. In dependency of the three above named possibilities [3] using specific probabilities.

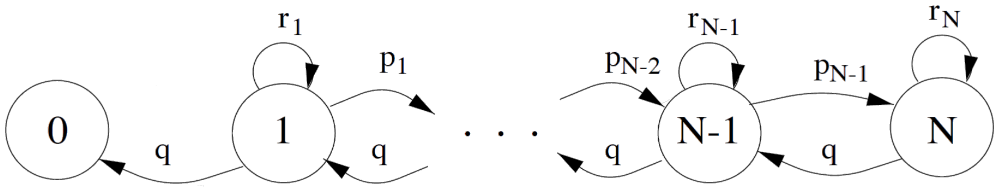

Figure 1: *A simple discrete Markov-Chain is displayed here. The number of states N give the nominal capacity of the battery in discrete units. One charge unit corresponds to the specific amount of energy needed for a transmission of a message. q gives the propability, r the propability that lets the chain stay in its current state and p indicates the recharging. Taken from [3].*

A more advanced model is presented in [4, 5]. In addition to the before displayed model (Figure 1), more degrees of freedom are considered in this advancement, shown in Figure 2. The most important change lies within the treatment of states and how much capacity they represent. While in the before described model a state correspond with the smallest energy consumption of the consumer, in the advanced model a more general approach is chosen. A state represents the smallest Energy Unit (SEU) the model could theoretically handle. This results in the fact that it is likely that more energy at one time is used by the device than one state can represent. This can be visualized when states are skipped, because the energy depletion can be seen as a number n states going the Markov-Chain down. The advanced model considers this case. It is also important that the likelihood for the recovery process p is not set by a constant parameter, but is set as a state-variant variable. The same holds for the possibility r, which indicates the battery remains in the same state over the period. For the depletion, different constant parameter considering the size of the jump between N states are set.

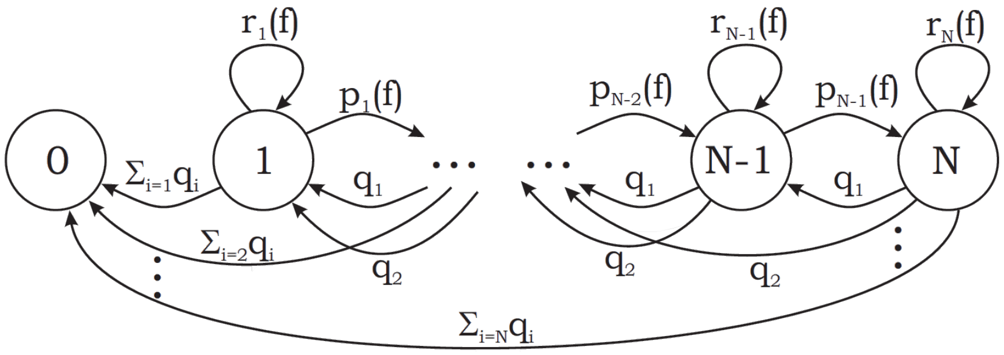

Figure 2: *The extension of the Markov-Chain in Figure 1. It can be seen that additional arrows mark the new possibilities to drain more than one state. In every state the probabilities p for recovery, q for transmission and r for being in idle are invoked. Taken from [5].*

Especially for the following adoption, the probability for the recovery process is important. Therefore, (1) shows the state depended probability for the recovery probability $p_j(f)$.

$$p_j(f) = \begin{cases} q_0 e^{-(N-j)g_N - g_C(f)} & j = 1, \dots, N-1; f = 1 \\ q_0 e^{-(N-j)g_N - d_f g_C(f)} & j = 1, \dots, N-1; f = 2, \dots, f_{max} \end{cases} \tag{1}$$

q_0 describes the basic probability for recovery of a state. It is affected by various parameters implemented in the exponential expression. In this equation N is the above introduced overall number of states representing the nominal capacity of the battery. To model the different behaviours of the recovery in dependency of the actual capacity of the battery, j is introduced and gives the actual state of charge.
The parameters g_N and g_C are taken the chemical behavior of the battery into account and controll the actual behavior. g_C the defined as *"Exponential decay coefficient; it is related to cell conductivity."* [5] and g_N *"Exponential decay coefficient; it is related to cell potential drop during the discharge phase."* [5]

As it can be seen in (1) g_C is not a constant factor, but a function related to f. Within the recovery, f is the iterative quantity how much discharge phases were passed before a recovery phase. With a growing f the recovery process is less likely to recover all of the energy drained from the battery in these phase. This is due to the fact, that a small depletion is more likely to get recovered than a huge depletion. It is important that $0 \leq f \leq f_{max}$, because it is not possible to recover more in the idle period that was drained in the period before.
In addition, the presented distinction of cases in (1) is related to the previous time of the discharge phases. If more than one discharge phase were passed by the device the time of these periods d_f is also an important parameter. The more time the device were in usage the less likely is a wide-ranging recovery. If there is only one time period of

discharge this factor is not important.
Some examples for actual implementations are presented in the following.

Modification of the stochastic model

In [7] parameters were chosen as close as possible to real batteries. They used a partial differential equations to calculate the required number of states N for a typical Li-Ion battery with 4.3 V nominal voltage and a rated currency of 125 mA. 1 000 000 chargeunits were used for their simulation. Thereby is q_0 the probability of an discrete idle slot, which means that a recovery can take place. For simplicity reasons they set $g_N = 0$. Thus, the impact of the current state of the battery is neglected. No further information about g_C and f are provided.

Another approach was done by [8], introducing the Markov-Chain in a Random Walk Process to model battery discharge, which will not covered here. The main difference in comparison to [7] is that the parameters were not directly set to a real case battery. To nominal capacity is described with $N = 100$. For every transmitting or receiving action of the device the battery can get drained with exact one state. As result, the battery has a life time of $\geq N$ operation intervals. Different between both papers is also the expression of the exponential term. In comparison to Chiasserini & Rao, [7] neglected the coefficient g_C and f. However, the interpretation of q_0 from both papers is similar and set to 0.1.

The third paper [9] uses the advanced model, as presented in [5], after some modification. The aim was to investigate battery properties for optimizing the battery life time. If the battery has enough remaining charge and remaining capacity left, the probability is expressed as exponential term. The difference is that no probability q_0 is set up, because idle phases are given from extern and not by stochastic processes. In [9] g_N is set as constant parameter and N is simulated with 200 charging units, which shows homogeneous compared to the papers presented before. However, $g_C(f)$ is configured as step function *"which decreases as the remaining capacity of the cell increases"* and is described as

$$g_c(f) = \begin{cases} 15.6 & 0 \leq f \leq 5 \\ 0.8 & 5 \leq f \leq 100 \\ 0.0025 & 100 \leq f < 195 \end{cases}$$

With the above introduced projects it can be seen that various variations were done to align the model to the research project. The scope of the project presented here, was to implement the model as close as possible to the basic approach from Chiasserini & Rao, but nevertheless be able to simulate the given wireless network.
In this project only the recovery is a stochastic process. Both the depletion and the idle period of the battery are not stochastic, but deterministic and given from the project framework. Another important factor is that the original model does consider background

drain of the battery. In idle phases the arrows revert back to the current state. In this modified project a constant background drain will be taken to account. The arrows with probability r and q will therefore be neglected. This changes can be seen in Figure 3. The arrows going up the modified Markov-Chain are marked with the probability p. All other arrows going down are invoked by conditions external given by the project framework.

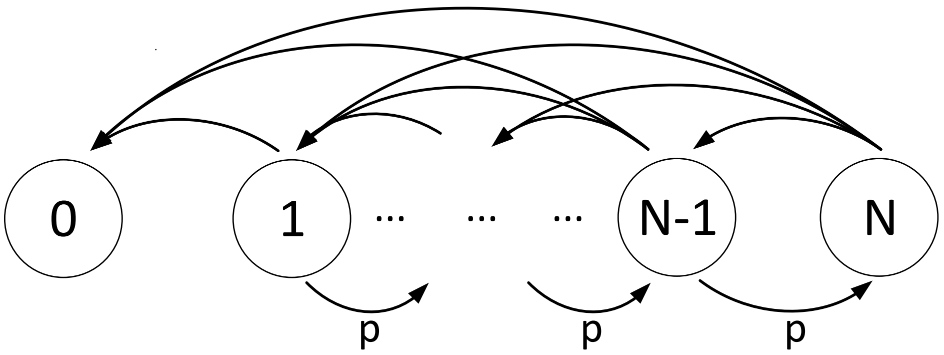

Figure 3: *Modified Markov-Chain aligned for the present project. The chain is reduced to one main probability p for recovery. No probability for a transmission as well as the an idle phase is needed, because exploit charge units from battery is an deterministic process given external from the project framework.*

As result the stochastic approach is reflected by the probability p for recovery. Similar to the others the original formula is states in [1] with

$$p_j(f) = q_0 e^{-(N-j)g_N - g_C(f)} \tag{2}$$

For this simulation it is modified to

$$p_j(f) = q_0 e^{-(N-j)g_N - g_C f} \tag{3}$$

The basic probability is denoted with p_0 as a constant parameters which gives the probability that a recovery process will start. The exponential term can be described in two parts. The first part $e^{-(N-j)g_N}$ describes a capped exponential growth in dependency of the overall charge level. The second part models an exponential decay in dependency of the consumed energy portion and is denoted by $e^{-g_C f}$.
The definition for N and j are simular to the presented papers and will not be changed here. N is the total nominal capacity expressed in SEU, while j is the current available capacity of the battery.
It was not possible to retrieve concrete values for g_N and g_C from the above stated paper. However, with the knowledge from the research done, this project will use constant parameters. In addition, the modification for g_C lead to an not inherent function in dependency of f. To model this dependency, g_C is also a constant parameter and will be multiplied with the current number of recovered units f within one recovery process. For the analysis and the concrete values of the parameters see section 3.3.

3 Adaptation of the model in OMNeT++

The implementation of the model was been made in the simulation program OMNeT++. The focus thereby lay in the behavior of the battery itself. A framework was designed as a simple communication network, where a sender and a receiver exchange message of known length in set, constant time intervals. Both parts of the communication chain deplete there respective batteries during this process.

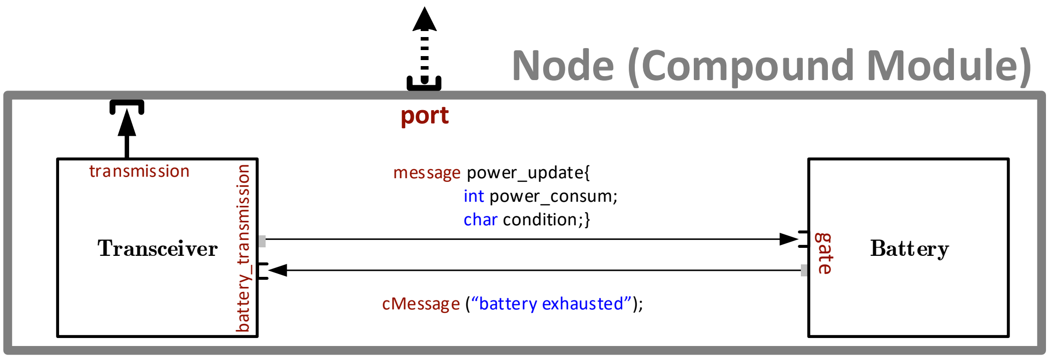

Figure 4: *Schematic of the used compound node. In the network two of these communicate with each other. The node contains a transceiver module, which is able to generate a message that is send over the network, as well as receiving it, in dependence of the initialisation. This communication goes over the transmission gate to the port of the Node. The two parts communicate with each other over the displayed ports with the shown message. The transceiver sends the battery the information how much power is consumed and what the current state is, whenever this changes.*

Within OMNeT++ the structure of the implementation is organized as follows:
A schematic is displayed in Figure 4. The communication network contains two of the shown compound nodes, one serves as a sender, the other one as the corresponding receiver. The message is generated in the transceiver module and send from its transmission gate over the compound node port.
The battery simulates its capacity level based on condition messages it gets from the transceiver. This messages come from the battery_connection gate of the transceiver and go to battery gate (called gate). The message contains the current status of the transceiver and its power consumption at this point. When the battery is empty this connection is also used the other way round and informs the transceiver that it has to stop processing now.
The message that is used for the empty-information is a standard cMessage from the OMNeT++ library. For the power update a custom message was created, which contains the power level as an integer and the current activity as a character. The message is send whenever the activity changes.

In the following the transceiver, as a framework, and the battery will be explained. In addition several parameters were set in the program to customise the simulation without changing the code itself. They include the core parameters of the model, as well as important framework characteristics, such as the used transceiver. They can be accessed through the omnetpp.ini and will be described as well. In this section also recommendations for the provided examples will be defined.

3.1 Framework implementation

To generate a framework for the stochastic model, which is simulated within the battery, the before outlined communication line is used. The message-sending (and therefore the energy consumption) is, as mentioned, not stochastic, but follows a deterministic pattern. This pattern is created in the transceiver, which simulates the transmission and receiving of a message with known length.
The transceiver module itself is able to handle the sending as well as the receiving. Within this scenario the two nodes are initialized for only one of the tasks, to generate a better controllable environment for the analysis.

The transmitting node is only active, when it is sending. The duration of this process is the product the length of the message and the data rate. This interval has a fix length as well as the interval between two sending processes, whereby both values are known. When the transmitter is not sending, it goes into deep sleep.
The knowledge of the sending interval is also important for the receiver. For this simplified framework it is assumed that the receiver opens a window for the message. Within it he is ready to receive the transmission. The receiving itself does, in this simplification, not require any time, because it should be approximated that the window is much larger than the actual receiving process. It is also assumed that no packages are lost in the channel. Therefore the receiving window always closes with the incoming message and no additional mechanisms in the receiver have to be considered. Outside the stated window the module goes into deep sleep, just like the sender.

The before described update of the power level is send out via the status message between the transceiver and the battery, whenever the status is changed. It does not update when the activity ends, because at this point another one starts (and therby triggers an update), so the message would be redundant.

3.2 Implementation of the stochastic model

The stochastic model described in section 2 is implemented in the battery module of the simulation. It structures roughly in three parts. First the consumption of the previous interval is determined. Secondly the recharge is stochastically computed. Lastly the incoming new status values are stored for the next call of the function, when the status is updated again.

The calculation of the consumption is relatively straight forward. The status and the power level of the last call were stored then. Furthermore a time stamp was set at this point so it is now possible to calculate the consumption of this interval with $c = \Delta t \cdot p$. The current capacity is reduced by this value.

Next the battery is able to restore some part of this consumed capacity, but only if it was a consumption from an highly demanding process like sending or receiving. If the module was in deep sleep it cannot recover the consumed capacity. This is the way the simulation deals with (not rechargeable) background drain.
This recharging is done as described within the theoretic model in section 2. Recharging is a step up in the Markov-Chain and while the draining can step multiply states, the recovering has to be calculated for every state with the exponential function (3). Hence the program needs a loop over all states that were going down during the consumption. Within this loop every instance has to call the exponential function. The result of this calculation is the probability to recharge this particular state. To check this instance a random integer between 0 and 1000 is drawn, multiplied with 0.001 and if the calculated probability is smaller than the result, a SEU is restored.

If the energy drain pulls the state to zero (or is high enough to deplete it even further) the battery calls itself empty and does not process any further. No recovery happens at this point, even when it could (theoretically) lift it above zero. When this happens the battery sends a message to the transceiver that it can stop processing. This way the simulation is stopped, when both nodes are empty.

To evaluate the data externally the build-in data collection possibilities of OMNeT++ are used. Three parameters are tracked for both nodes:

- total capacity
- consumption (in the respective instance)
- recovery (in the respective instance)

All values are converted to [As] before they are stored, even if the calculations are processed in [SEU]. The consumption alternates only between the active and the passive state of the node and is otherwise time constant. Nevertheless in the graphic output it allows a good comparison to the recovered capacity.

3.3 Parameters of the simulation

The simulation depends on several parameters that lay within the model itself or define the framework. All of them can be accessed over the *omnetpp.ini*.

The framework parameters pertain the transceiver and its performance. Four examples for the transceiver are given and can be seen in Table 1. The power consumption of the receiver depends on the window that it opens. The size can also be set in the *omnetpp.ini*, the default value is 0.95, means the node is active 5 % of the time.
It is also possible to enter own values. Another important parameter of the transceiver

is the battery capacity. This impacts the runtime of the simulation and will be evaluated in an instance.

	CC2530	**ESP8266**	**CC2650**	**RFD22301**
data rate	250 kbps	54 Mbps	1 Mbps	250 kbps
Deep Sleep	200 µA	10 µA	1 µA	4 µA
RX	24.3 mA	60 mA	6.1 mA	12 mA
TX	33.5 mA	215 mA	9.1 mA	12 mA

Table 1: *The four pre-set examples for the transceiver that can be accessed over the omnetpp.ini. Given are the data rate and the power consumption at deep sleep, sending and receiving. The table was given in the original task of the course.*

The most important factor of the model is the size of the Markov-Chain N. It has to be long enough that the SEU can cover even the smallest energy consumptions. In addition the results would be best, if within this consumption were enough states so that a stochastic process can be observed. To estimate a value for N the most economical consumer *ESP8266* is taken to account with a message only 64 byte long. It consumes

$$\frac{64\,\text{bit} \cdot 8}{54\,\text{Mbit/s}} \cdot 215\,\text{mA} \approx 2 \cdot 10^{-6}\,\text{As} \tag{4}$$

A small commercial coin battery contains between 30 mAh [10] and 620 mAh [11]. In reality probably a larger battery could be used, but at this point a capacity of 620 mAh $\approx$ 2000 As should be considered. This is the capacity the battery holds (according to manufacturer specifications) before the nominal voltage drops below the boundary of 2.0 V. Below this the transceiver does no longer operate reliable.
This yields as a ratio of

$$\frac{2000\,\text{As}}{2 \cdot 10^{-6}\,\text{As}} \approx 10^9 \tag{5}$$

So the number of smallest energy units (SEUs) and thereby N should *at least* be 10^9, better more. The default value is set to 10^{10} and should be handled with care. A larger N leads to more accurate results, but significantly increases the simulation time.

As elaborated before, the key factors of the stochastic model lay within (3) and are q_0, g_N and g_C. Derived from N they can now be determined. As explained earlier g_N and g_C are exponential factors, while q_0 is just the basic probability, that a SEU is recovered. g_C is part of the decay function that is controlled by f, which counts how much capacity was consumed in this cycle. The specific value for g_C depends on the used battery, but it should at least be prohibited that the whole capacity of the battery can be restored in one turn. In the default scenario it is assumed that 1 % of the maximal capacity can be recharged (under neglect of all other influencing factors) in one cycle. A good approximation for this assumption is a factor of $g_c = 10^{-8}$. That means that the 10^8th SEU has a $1/e \approx 37\,\%$ chance to recover. For every following state the probability decreases. g_N on the other hand is responsible for the overall dependency of the charging state. It

regulates that the battery recharges more capacity when is is emptier. For this parameter a strong influence is wanted to bring out this effect. For the example $g_N = 10^{-10}$ is chosen. A lower factor leads to more recharge, while a higher prohibits this. When the battery is half empty the probability (under neglect of all other influencing factors) to recharge is $p = e^{-(10^{10}-0.5\cdot 10^{10})\cdot 10^{-10}} \approx 60\,\%$. That means roughly, when the battery is above above half full it is less likely to recharge then not. Below half empty it is more likely to recharge than not and with decreasing capacity it gets (exponentially) more probable.
With this configuration the battery is very stable near its depletion, when a small consumer is used and it goes into a very quasi equilibrium that lengthening the simulation a lot. To counter that q_0 is set to $1 > q_0 > 0$ so the probability is never higher than this factor. In the given scenario it is set to 0.95.

When changing the core parameters, the elaborated dependencies of the parameters should be carefully taken to account.

Independent from the stated parameters, which concern the model and the framework, there is one OMNeT++ inherent factor to consider. The program is by itself discrete and handles time with a fixed-point 64-bit integer. This controls the resolution and the maximum runtime of the simulation as seen in Table 2.

Exponent	**Resolution**	**Approx. Range**
-18	1 as	± 9.22 s
-15	1 fs	± 153.72 min
-12	1 ps	± 106.75 d
-9	1 ns	± 292.27 a
-6	1 µs	± 292 271 a
-3	1 ms	$\pm\ 2.9227 \times 10^8$ a
0	1 s	$\pm\ 2.9227 \times 10^{11}$ a

Table 2: *Possible exponents for the fixed-point 64-bit integer simTime_t in OMNeT++, as well as the thereof resulting range and resolution. From the OMNeT++ Simulation Manual.*

By default the exponent is set to -12. This might lead to an overflow of the simulation time for energy efficient transceivers with a long deep sleep phase between the sending intervals after 106.75 days. But on the other hand the resolution should not be set to low, to make sure it can resolve even shortest sending events.
As new value for the simulation resolution was set to ns. This should be sufficient for every occurring event and enough time for even long lasting components.

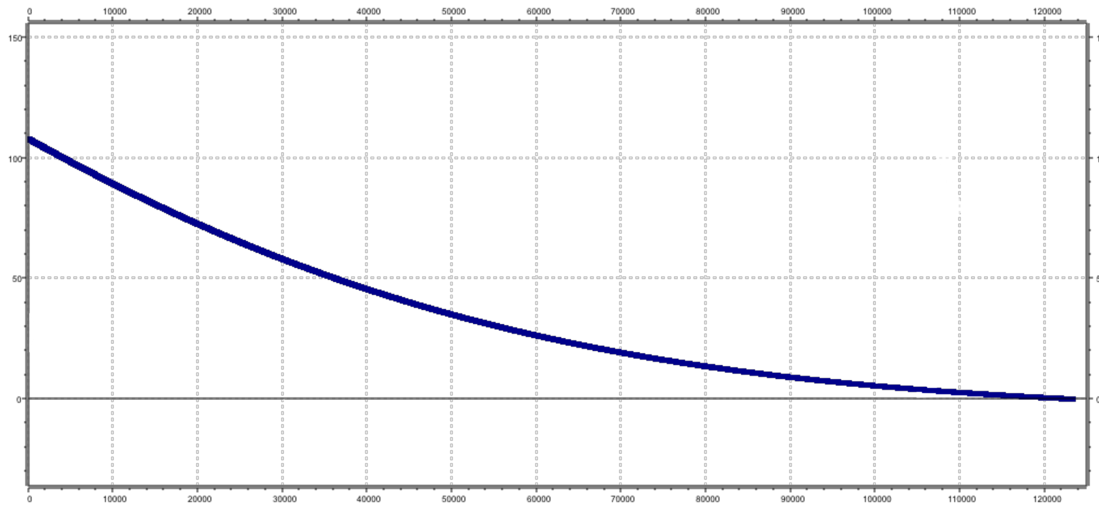

Figure 5: *The battery capacity over time using the ESP 8266 WiFi module which consists a transceiver and micro controller. Parameters were set to the following: sending interval* 3 s*, payload of the messages of* 1000 bit*. The number of remaining* As *of the battery are shown on the y-axis, the time is plotted in seconds on the x-axis.*

4 Analysis and limits of the model

Several simulations with different receivers and transmission parameters were run with the presented model. In the following some of the results will be presented to show key elements of the model.

It is trivial to see that the exponential behaviour for the recovery process is greater the more energy the transceiver needs for sending respective receiving. Within Table 1 the ESP8266 needs the most from the given examples. The energy consumption from Table 1 describes both the transceiver and the micro controller for this WiFi module. To ensure a high need for energy in the following examples the Payload of the messages is set to 1000 bit and a sending interval of 3 s is used.
For the first results, this project focus on the receiver side. In Figure 5 the battery capacity is shown. For the above stated parameters the depletion of the battery follows a weak exponential decay. This is because the emptier the battery capacity gets the stronger the recovery evolves, which can be seen in Figure 6. However, the stochastic process of the recovery is caped to the maximum of the depletion in the period before. This leads to fact that the capacity will never be greater than in the period before. As result, the recovery process leads to a plateau with a nearly stable regain rate.

In the shown simulation the life time of the receiver, limited by the battery capacity, was 121.4 d. After that the battery was depleted. Nevertheless the sender kept sending, what is not shown here for clarity reasons. In fact, the difference of life time is quite big, because of the different usage of the battery on both side of the transmission.

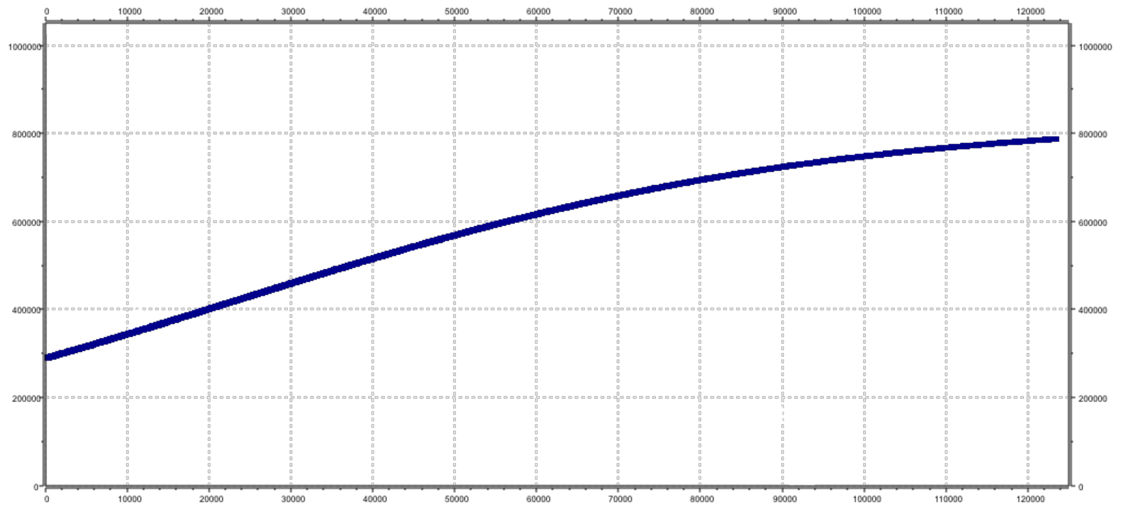

Figure 6: *The recovery of SEUs is shown in this Figure. The parameters are similar as in Figure 5. The time in seconds is plotted on the x-axis, the number of recovered SEUs on the y-axis.*

To demonstrate that difference in life time, another simulation should be shown. The results of a run with CC2530 transceivers can be seen in in Figure 7. It can be seen that the life time of the battery for sender and receiver exhibits considerable differences. The life time of the sender is approximately three times longer than the transceivers. In addition, the influence of the exponential recovery probability is clearly stronger for the receiver than for the sender side. This is due to the fact that the receiver has a longer on-time than the sender and the energy consumption is in general higher by transmitting than receiving. When the consumption is small the most of the time nearly all of the states are recovered in each cycle, because the influence if the second half of the exponential term is nearly negligible. When it kicks in for higher consumption also the effect of the first part of the exponential equation has a stronger impact.

Last but not least, the stochastic characteristics can be exposed by looking at the recovery rate of a sender. An example is shown in Figure 8. The parameters were set to a sending interval of 40 s and an data payload of 100 bit. In combination with the sending speed of 54 Mbit/s of a ESP8266 module the transceiver only uses small number of charging units. This results in a high resolution of the stochastic process.

In comparison to Figure 6 the recovery can not be described by one curve, but as a stochastic process with similarities to a Random Walk. However, this effect also happens within the other graph is is just less visible.

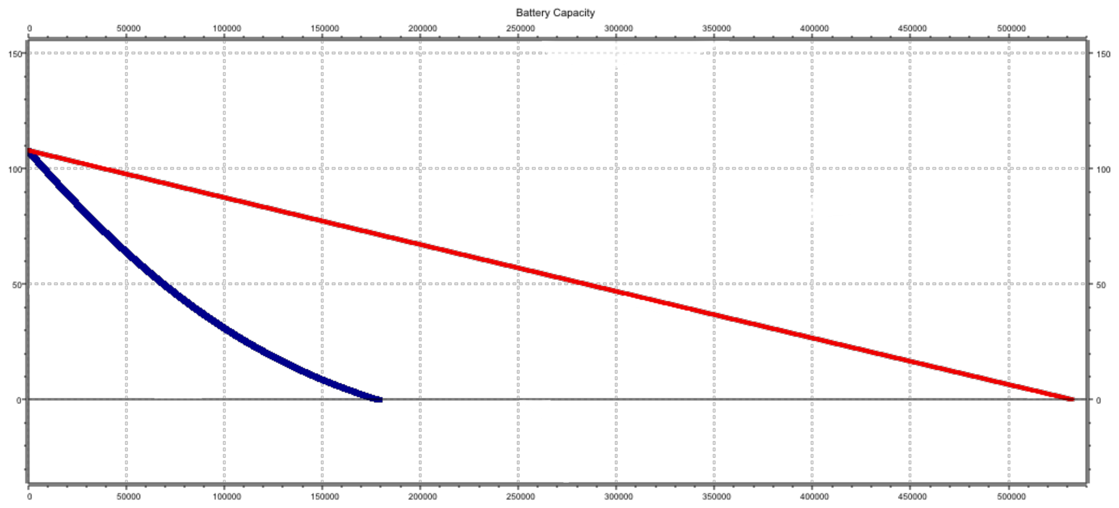

Figure 7: *The comparison of the battery capacity between sender and receiver is shown here. A CC2530 transceiver was used with the following parameters: sending interval* 5 s*, payload of the messages of* 100 bit*. The number of remaining* As *on battery side are shown on the y-axis, the time is plotted in seconds on the x-axis.*

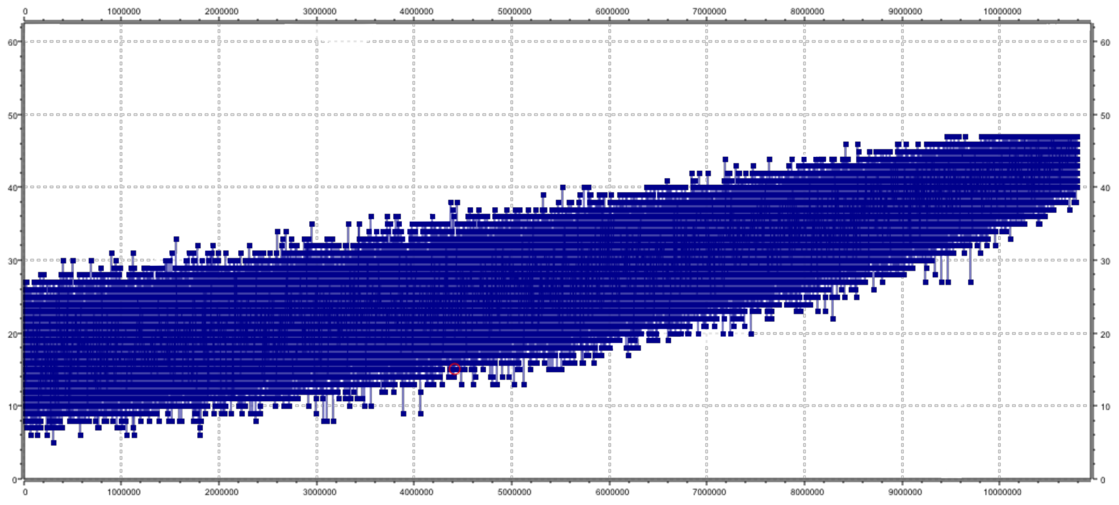

Figure 8: *The stochastic characteristics of the recovery are outlined here. For this diagram an ESP8266 module with the following parameters were chosen: sending interval* 40 s*, payload of the messages* 100 bit*.*

Limits of the project

As described above the simulation time was in a range of one 100 days or above. This led also to a great simulation time, because a transmission is done within a range of seconds and so a large number of iteration has do be simulated. At this point this project considers and evaluated a trade-off between the precision and resolution of the project and the maximum simulated time. Especially for transceiver types like ESP8266, drained energy per action is small. The increasing life time of the battery led in early simulations to an overflow of the simulation variable *simtime_t* and a cancellation of the simulation. Hence, *simtime_t* had to be modified. In addition, it should be recognized that a simulation can take several hours until result can be retrieved.

To reduce the simulation time in this approach only a minimum transmission model was considered, which only contained a sender and a receiver. This was sufficient to receive basic analysis data. A continuing process would be to model a network with more than two transceiver and let them switch tasks between sending and receiving.

A third limitation was the transmission itself. A constant, time-invariant and completely ideal channel underlies this model. Messages and data packages were transmitted without an delay and an error-probability of 0 %. An extension would be to cover also this parameters. It can be assumed that the transceiver will use more charge units of the battery than before for the same type of transmission.

It should also be noted that this project considers a discrete model. One effect is, that the recovery is no process over time. Instead, the recovery is modelled instantaneous after the device drains charge units of the battery and drainage and recovery happen in the same instance. Hence, the shown recovery process is not a realistic process compared with the real battery. However, the process should lead to the same result, if the parameters introduced in Chapter 3.3, are considered as correct.

At the end, the the amount of charge units were a restriction of the model. In comparison to implementations of other publications presented in Chapter 2 this projects uses a much larger number of states. This was because the resolution of the charge units belongs to the minimum allowed energy consumption of the transceiver. One of the consequences is the limitation of the battery capacity to ensure a feasible simulation time. Therefore, it must be considered whether the underlying concept by Chiasserini & Rao is the best suited for modelling battery depletions for devices with low energy demand.

5 Conclusion

In conclusion it can be said that implementation of the stochastic model from Chiasserini and Rao is possible within OmNeT++ and leads to plausible results. For low energy wireless networks however it is a quite heavyset simulation.

The model itself contains heavily on the inherent parameters. To find fitting values an appreciable amount of understanding for the model is necessary. Especially, because the exponential factors are highly attached with the number of states within the Markov-Chain. Improper values result in mismatching results and either a linear drain or virtually no consumption and thereby an everlasting battery.
This severe dependency means that for every new setup that derives from the pre-set build all parameters have to be checked, not just the size of the model itself. Additional work could be done to validate better fitting parameters for existing batteries.

Regardless this challenges to adjust the model, the basic needed functionalities can be observed. A recovery process takes place and is observable. It is changes with the overall level of the battery capacity, as well as the amount of drained energy per consumption cycle. In addition the recovery is clearly stochastic, a key characteristic of the model.

Within the given framework of a wireless communication chain ,with its power efficient consumer, the simulation needs a lot of runtime. This is nearly exclusively caused by the exponential function that has to be called for every recovery step in the Markov-Chain. At this point the authors see no way to improve the performance without modifying the underlying model.

As mentioned in the introduction this project was part of an university course where different models, as described in [1], were implemented in OMNeT++. Without having taken direct comparable measurements, it seams that other models are considerable faster and simulate simple communication lines within seconds or at most minutes.
This leads to the conclusion that the model of Chiasserini and Rao might not be the best for the given scenario, but be designed with other fields of application in mind.

References

[1] M. R. Jongerden and B. R. Haverkort, “Battery modeling”, Centre for Telematics and Information Technology, University of Twente, 2008. [Online]. Available: `http://doc.utwente.nl/64556` (visited on 02/08/2017).

[2] C.-F. Chiasserini and R. R. Rao, “Pulsed battery discharge in communication devices”, in *Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, ACM, 1999, pp. 88–95. [Online]. Available: `http://dl.acm.org/citation.cfm?id=313488` (visited on 02/08/2017).

[3] ——, “A model for battery pulsed discharge with recovery effect”, in *Wireless Communications and Networking Conference, 1999. WCNC. 1999 IEEE*, vol. 2, IEEE, 1999, pp. 636–639. [Online]. Available: `http://ieeexplore.ieee.org/abstract/document/796721/` (visited on 02/08/2017).

[4] ——, “Improving battery performance by using traffic shaping techniques”, *IEEE Journal on Selected Areas in Communications*, vol. 19, no. 7, pp. 1385–1394, 2001. [Online]. Available: `http://ieeexplore.ieee.org/abstract/document/932705/` (visited on 02/08/2017).

[5] ——, “Energy efficient battery management”, *IEEE journal on selected areas in communications*, vol. 19, no. 7, pp. 1235–1245, 2001. [Online]. Available: `http://ieeexplore.ieee.org/abstract/document/932692/` (visited on 02/08/2017).

[6] T. Crompton, Ed., *Battery reference book*, Third Edition, ser. Battery journals, trade organizations and conferences, Oxford: Newnes, 2000, ISBN: 978-0-7506-4625-3. [Online]. Available: `http://www.sciencedirect.com/science/article/pii/B9780750646253500688` (visited on 02/16/2017).

[7] D. Panigrahi, C. Chiasserini, S. Dey, R. Rao, A. Raghunathan, K. Lahiri, *et al.*, “Battery life estimation of mobile embedded systems”, in *Fourteenth International Conference on VLSI Design*, IEEE, 2001, pp. 57–63. [Online]. Available: `http://ieeexplore.ieee.org/abstract/document/902640/` (visited on 02/15/2017).

[8] Q. Ge, R. Chandramouli, V. S. Swaminathan, and S. V. Desai, “Stochastic analysis of unattended sensor battery life time”, E. M. Carapezza, Ed., Apr. 3, 2008. [Online]. Available: `http://proceedings.spiedigitallibrary.org/proceeding.aspx?doi=10.1117/12.782934` (visited on 02/15/2017).

[9] M. Adamou and S. Sarkar, “A framework for optimal battery management for wireless nodes”, in *INFOCOM 2002*, vol. 3, pp. 1783–1792. [Online]. Available: `http://ieeexplore.ieee.org/abstract/document/1019432/` (visited on 02/15/2017).

[10] Energizer Holdings, Inc., *Data sheet CR1025*. [Online]. Available: `http://data.energizer.com/pdfs/cr1025.pdf` (visited on 02/16/2017).

[11] ——, *Data sheet CR2450*. [Online]. Available: `http://data.energizer.com/pdfs/cr2450.pdf` (visited on 02/16/2017).