3. Problem statement/research question

The research question presented here is simple:

Can the practical implementation of the presented self-stabilization algorithm guarantee the reliable FIFO message delivery over bounded non-FIFO and duplicating channel?

Using the central research question as a guide throughout this study, we can clearly identify the main goal we have to work towards: gathering the necessary data from the simulations and experiments on the implementation, to establish sound and quality proof that indicates the correct working of the algorithm according to the hypothesis presented in the original paper.

3.1 Objectives

The research question presented above translates in a set of clear objectives. These have to be met in order to have sufficient empirical data for the hypothesis of the original paper to be confirmed as true.

- Implement the algorithm in TinyOS. In doing so I have the full freedom available to me, to make my own choices in terms of coding strategies. The only requirement is that the original algorithm is at least implemented as is presented. When different possibilities present themselves for me to implement a certain behavior of the algorithm, I can make the choice myself. It is therefore my responsibility to make sure that the choice I make is well thought out, and doesn't compromise on the deliverables that are set out in these goals.

- Establish good performance variables and benchmark the algorithm by performing simulations in function of the predefined performance variables.

- Optimize the algorithm by adjusting the performance variables. These adjustments should be made based upon the data and results that were acquired from the simulations.

- Provide the necessary proof of the self-stabilization concept by using the LibReplay tool to insert, duplicate and reorder packets during communication.

- Write the report, keeping in mind that future researchers should be able to continue or expand on my work in case they want to implement the algorithm in a bigger application or adjust based on new information.

3.2 Challenges

During every empirical research study, you are bound to come across challenges which will sometimes require to adjust the planning or find a way around these challenges. I can't think of a field of study where this is more true than in Computer Science. The biggest obstacle in this study is without a doubt TinyOS itself.

3.2.1 Debugging TinyOS applications

> "Bug hunting in sensor networks is challenging: Bugs are often prompted by a particular, complex concatenation of events. Moreover, dynamic interactions between nodes and with the environment make it time-consuming to track and reproduce a bug[**?**]."

Although LibReplay (see Methods section) offers the ability to debug sensor network applications like in sequential programming environments, it is still not feasible to do this consistently for a lot of bugs during development. To use LibReplay for even the smallest bugs would require a lot of time. All this means that I had to search for other debugging methods during my work, such as responding to the compiler output and using printf statements[**?**].

3.2.2 TinyOS documentation

TinyOS was first released in 2000. Initially it gained a lot of momentum for being a very good low-power embedded OS, especially in research fields. Unfortunately it has fallen very quiet around the whole TinyOS ecosystem in the last five years. This makes the online documentation often contradicting because of different versions being used, and the up-to-date information is for a large part vastly spread out over the Internet in the form of GitHub issues and commits, TinyOS-help mailing list, random articles on blog sites etc.

3.2.3 Abandoned TinyOS support

TinyOS used to offer support through an email address (tinyos-help@millennium.berkeley.edu). This tinyos-help mailing list is still available to visit, but it completely unorganized and a total mess to go through.

To illustrate my point, I'm going to use a quote here that is taken directly from the offical TinyOS FAQ page. This is part of the wiki pages on TinyOS and was written by the TinyOS developers themselves.

"Generally, the best place to start is with the authors of the paper describing it. Even if they don't maintain an implementation, chances are they are aware of existing versions of it. Emailing tinyos-help isn't usually effective."[**?**]