Why and how to write the state-of-the-art.

State-of-the-art (SoTA) is a step to demonstrate the **novelty** of your research results. The importance of being the first to demonstrate research results is a cornerstone of the research business. You cannot get a Nobel prize (anymore) by learning Einstein’s law of photoelectric effect by heart and presenting it as your own. Einstein did it before you, and everyone knows it because he published it. When Einstein published his theory the theory had novelty. Einstein could demonstrate his theory’s novelty by presenting a SoTA and showing that no other researcher had ever presented such results. That’s why he got a Nobel prize and you will not.

Besides demonstrating the novelty of your research results, a SoTA has other important properties:

- It teaches you a lot about your research problem. By reading literature related to your research problem you will learn from other researchers and it will be easier for you to understand and analyze your problem.
- It proves that your research problem has relevance. If many people are trying to solve the same research problem as you, and if you can demonstrate this in your SoTA, then no one can tell you the problem you are trying to solve is not important.
- It shows different approaches to a solution. By seeing many different approaches taken by other researchers, you can evaluate your own approach and realize its novelty (or lack of it) easily. You can also see which approaches are the most popular and which are dead ends.
- It shows what you can reuse from what others have done. Especially when doing research on new software, it is amazing how many people have made the exact software you are planning to make. Just do a search on sourceforge and github.

So how to write a good SoTA? Writing a good SoTA is 110% dependent on having a clear problem definition. If you have failed in defining your problem clearly, you will fail in writing a good SoTA. The reason is that you will not know what related research you should investigate. So if you have problems with your SoTA, please go back and work on your problem definition! Here are some steps/hints on starting to write:

1. SoTA is not a one-way road. You will not sit down one evening and write your SoTA. You will do it all the time while writing your paper/report. Knowing what other researchers are doing should be a part of your life for all the duration of your research. So an important step is to create a system of registering and summarizing what you read. Use some bibliography software such as Mendeley, BibTeX or EndNode or Zotero, register everything you read, and register your understanding of what you read, in your own words.
2. Be critical when choosing your literature. Don’t read everything. There is a LOT of garbage out there on the web, and you don’t want to waste your time on garbage. One important criteria for choosing your literature is to make sure that it is peer-reviewed and is already presented/published in well-known conferences/journals. In case of technical IT-related stuff, ACM and IEEE are good places to start (do searches in Engineering Village). It is also a good idea to set up an initial SoTA literature list together with your supervisor.
3. Stop reading! Make an initial selection of literature (10-20 papers, depending on research problem) and stick to these for a while. Don't go on finding new papers all the time, or you will never finish your thesis!

4. Spend time on analysis and not on making summaries! A mere summary of 10-20 papers is not a SoTA. There is software out there that can summarize any paper for you, automatically and much faster than you ever will be able to. Your summaries become a SoTA only when you relate the SoTA papers to your own problem analysis.

5. Always give credit! Not giving credit for others' research is also called plagiarism.

6. For more advanced writers: It is always a good practice to document your methodology for doing state-of-the-art survey. This means you should document how you searched for literature, what literature you included and what you excluded, how you did your analysis and so forth. This is called systematic review, and a de facto guide for doing systematic reviews in the field of software engineering is available here. You can also find many useful links on wikipedia.