

In this class we discussed risk, ways to categorize risk, and ways to deal with it. We started with a review of some of the most famous problems with software that have been documented. Check out the following links:

- <http://www.devtopics.com/20-famous-software-disasters/>
- <http://www.google.com/#sclient=psy&hl=en&q=famous+software+bugs>
- <http://catless.ncl.ac.uk/Risks/>

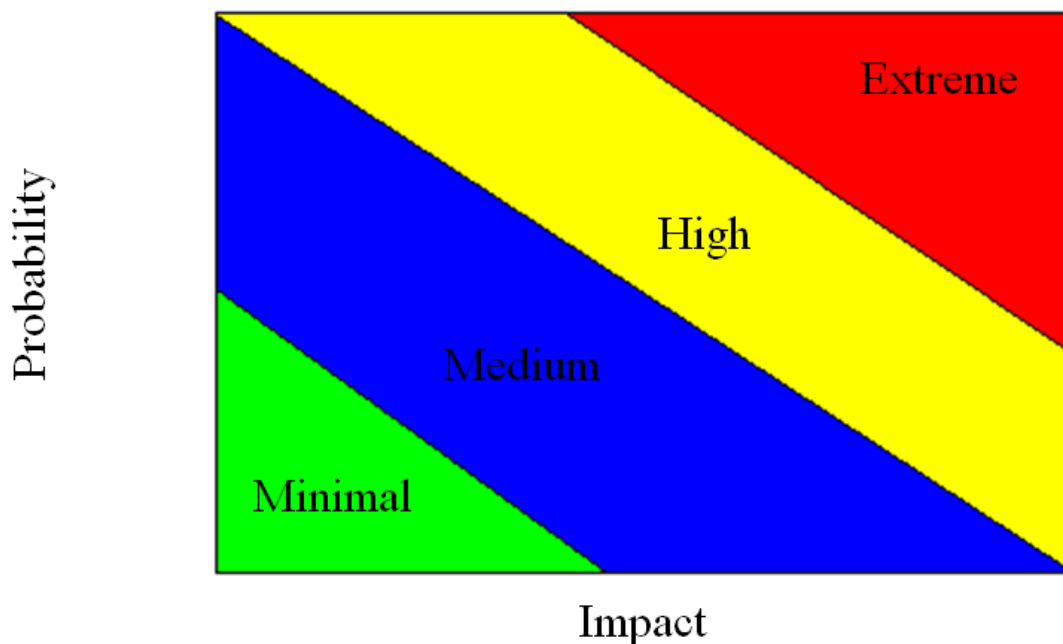
I like to use a really simple definition for risk:

Potential harm that may arise from some present action

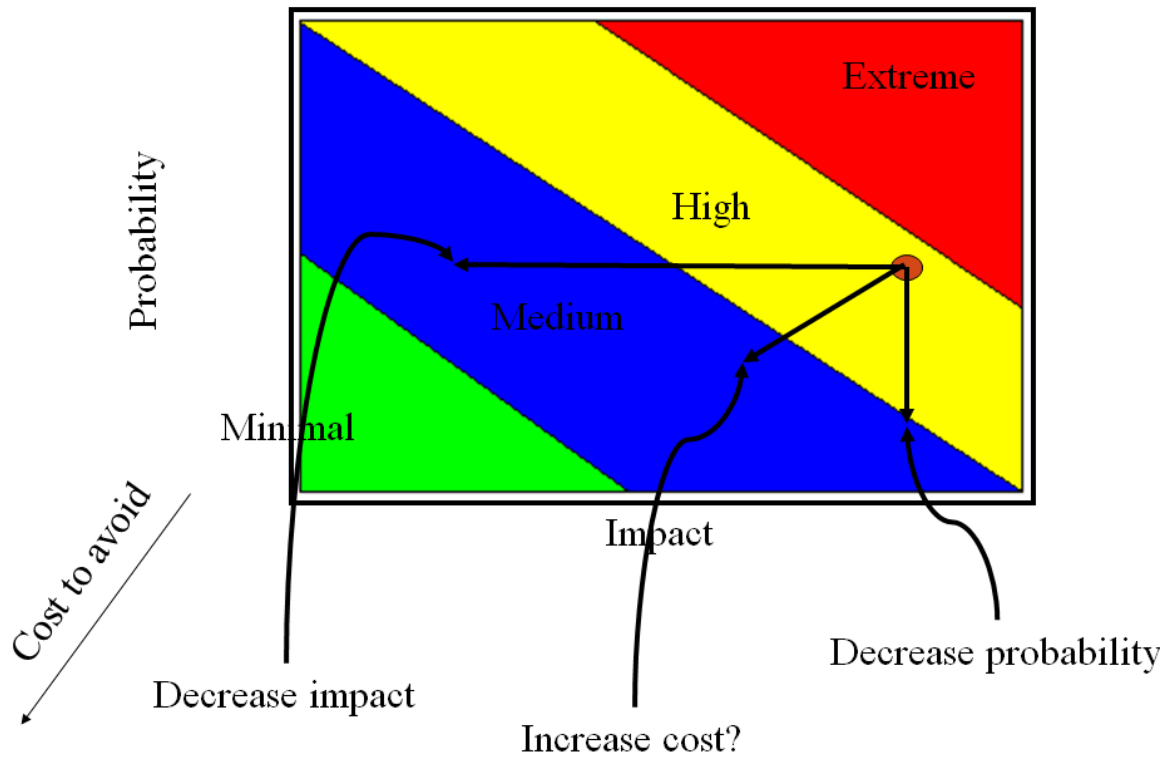
There is always the case that the harm may arise from inaction. We'll think about risk in terms of the following action plan (ACAT):

- Avoid
- Control
- Accept
- Transfer

But in order to really make sense of this planning we need a way to reason about risk. Consider the following diagram:



There may be shifting of the colored regions depending on your particular philosophy.



Then, if you consider mitigation strategies, and the cost associated with them, you arrive at a different model. Mitigation should be looking at ways to either decrease the impact (if the risk materializes) or decrease the probability of the risk occurring. The tradeoff is dollars spent on the mitigation. If you spend more to avoid the risk than what the impact of the risk would cost, you might have made the wrong decision.