For the antipatterns activities identified by the antipatterns test (part of this week's assignments), comment on how these might negatively influence specific requirements engineering activities (e.g. elicitation, representation, communication, etc.) Also, discuss how you will mitigate these challenges via refactoring.

Some of the antipatterns that the test identified based on my answers were “Leader Not Manager”, “Rising Upstart”, and “Autonomous Collective”. The first two are Management Antipatterns while the last one is an Environment Antipattern. In case you got different ones, I’ll summarize the description provided. “Leader Not Manager” is when leadership has a vision, but no plan. “Rising Upstart” is when “superstars” are willing to take the time to find the best approach to their job. Finally, “Autonomous Collective” is when employees are considered all the same, so there is no one leading and making decisions. Each one of these can be detrimental to the requirements engineering process.

Let’s start with “Leader Not Manager”. This antipattern may make requirements analysis and agreement more difficult because they may try to turn every raw requirement into refined requirements that contradict each other or are just wrong because they are more concerned about having some grandiose system, but don’t stop to consider how each requirement could be implemented or if it even fits within the scope of the system. One way to refactor this antipattern could be to ask questions like “Wouldn't X be impossible if we do Y?” or “Does X make sense if the goal of the system is to Z?” This way, you can make your manager take some time to really think about the requirements rather than just being a “yes-man” to the customer.

“Rising Upstart” can be frustrating because it may lead to having to do more work in the long term. This may be because this type of person might be good at what they do, but don’t think solutions all the way through, which might make validation difficult. They might be quick to a solution, but use the wrong data or maybe even just skip over validation in an effort to get “more work” done. Poorly executed requirements slow down the development process. This can be refactored by having another team member specify the validation criteria first. This way, the individual can move as fast as they want, but it must meet the existing validation criteria before it can be considered complete.

Finally, “Autonomous Collective” can make RE processes, such as elicitation, difficult because this type of environment leads to lack of motivation. Lack of motivation means that the RE who is gathering requirements might not dig deep to find valuable requirements. They might just take what the customer wants at face value , but sometimes the customer is wrong. To refactor this type of environment, you need to be the individual who is willing to put in a little extra effort. Having just one person who takes charge can provide direction for the team or project, thus resulting in more efficient development of the system.