The theory that workers can be motivated by a combination of rewards and punishments. Whether this can work for repetitive factory work is debatable. But even if it does, software developers are not factory workers. Once the basic requirements of adequacy and fairness have been met, external rewards cease to be a strong motivating factor. In fact, in some cases they can even diminish motivation.

It can be intrinsically rewarding to perform a task you enjoy, like crafting well-architected micro-services or finding a particularly elegant solution to a design problem. But once the external reward becomes the reason to perform the task, it can start to take over and kill the intrinsic enjoyment.

The purported solution is to ensure intrinsic motivational factors are acknowledged and included. A growth framework cannot be solely about deciding how much someone is paid and what their title is. It must also expose and elevate internal motivational factors.

There is a growing resistance from some quarters to the idea of having a ‘architect’ or ‘tech lead’ dictating technical choices from the top-down. This contradicts autonomy, which reduces intrinsic motivation. Rather, with autonomous teams and micro-services, we can try to create an environment where good architecture and decision making is everyone’s responsibility.

A software engineer’s goal is not to write code, it’s to [build product](https://medium.com/@coderdan/the-coders-role-92fbc6cd5f2a).

A software developer’s role is to solve problems for users.

**Growth frameworks are separate from titles.** It is possible to separate the concern of external-facing titles from skill and growth. The most important question to answer is “How do I grow?”

**Autonomy, mastery and purpose are crucial.** The three work in concert to produce a workplace where people feel valued, want to contribute and are able to contribute.

<https://medium.com/better-programming/career-growth-frameworks-in-software-engineering-a-review-4aa6c59a9cf6>