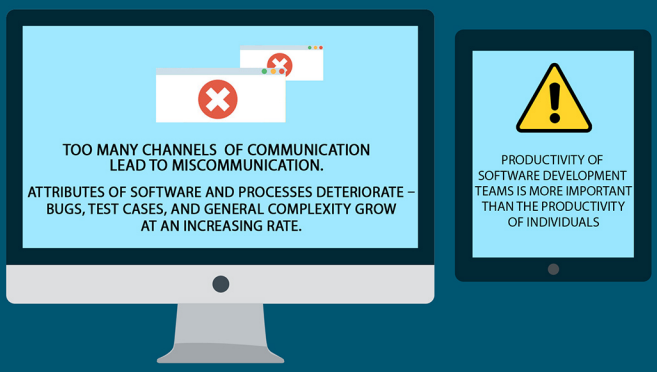
PRODUCTIVITY IN THE SOFTWARE WORLD



Software is delivered by teams – made up of individuals with diverse skills, backgrounds, and strengths + weaknesses

METRICS TO MEASURE INDIVIDUAL PRODUCTIVITY

Lines of tested, production-ready code produced. Due to reasons ranging from the difference in verbosity of languages and varying quality of code per individual, this metric doesn't work well.

METRICS TO MEASURE SOFTWARE TEAM'S PRODUCTIVITY

Techniques counting Functions Points, Feature Points, COCOMO, and more – are no longer in use.
The new metric is 'Velocity' which is the team's productivity, calculated by measuring the number of story points delivered to production during a 'sprint'.

SOFTWARE DEVELOPMENT

Being efficient ✗

Being effective ✓

Factors that affect the effectiveness of Scrum/Agile sprint-based processes

INTERRUPTIONS

- ✗ Teams and individuals are interrupted for status requests from multiple stakeholders
- ✓ Teams should ideally regroup at the end of the day to re-assess the status and communicate with the clients

SCOPE CHANGES DURING A SPRINT

- ✗ Scope changes hit productivity twice over – some team members need to stop working on what was planned, switch the context, and learn the new requirement/change
- ✗ Under pressure to deliver, developers might take shortcuts using poor practices, which will add technical debt
- ✓ For Agile/Scrum models to work and be able to consistently deliver – every sprint should be scope boxed – no changes to requirements and no new stories

LACK OF AUTOMATION

- ✗ Despite the availability of automation tools, software development teams and managers refuse to take this up seriously
- ✓ Adoption of a static code analyzer, automation unit and integration along with system-level testing are most important technical initiatives that can improve productivity

DIFFERENCES IN SKILL LEVELS OF DEVELOPERS

- ✗ Poorly skilled individual contributors reduce the productive time of the skilled ones
- ✓ Removing an underperforming developer can increase the team's productivity

TECHNICAL UNKNOWNS

- ✗ Using multiple third-party libraries with potential poor documentation, compatibility issues caused technical unknowns -- the impact is discovered late in the development cycle
- ✓ Developers should anticipate the risks, be prepared with alternatives, and provide adequate buffers for POCs during estimation
- ✓ Developers must also upgrade themselves on a regular basis to avoid technical unknowns

✓ To increase productivity, every organization must fine-tune its practices ranging from hiring and engineering processes, as well as by fostering a culture of excellence ✗