eficode

GUIDE

How to estimate DevOps toolchain maintenance and support costs

We're so used to talking up the benefits of automation that we often overlook the costs. How much valuable time are we spending on toolchain maintenance, and is there a better way to manage it?

Running an effective and highly optimized software development and delivery organization is a demanding job. Teams have to balance tight schedules, ever increasing system complexity and a scarcity of skilled experts.

An integrated DevOps tool chain, like any assembly line, is a key enabler for operational efficiency. As the level of automation increases, the tooling platforms become more extensive and the accompanying maintenance and support tasks become more complex.

The combination of demanding schedules and increasing system complexity forces teams to choose where to focus. Should they be the experts in the software development and delivery, maintaining the assembly line tooling, or both?

For most organizations the answer should be relatively clear. They should spend the majority of their time working on the software itself.

The assembly line is a must-have system but the teams rarely need to excel in tool maintenance to be competitive in the field where they actually compete. It is an area that can be outsourced to domain experts.

You can easily find out the costs related to an outsourced managed assembly line service. However, for an educated "make or buy" decision, one also has to be able to estimate the internal costs of running an internally maintained tool chain. The purpose of this document is to give guidance for making these estimates.

This guide does not provide any actual numeric information. The actual figures depend heavily on the size of the organization, the number of tools and toolchains in use, and the skills and professionalism of the internal support and maintenance operations.

Basic cost factors

Overview

The costs of running internally managed tool chains are often hidden because the assembly lines are set up and maintained by project personnel as a side job. Additionally, while lacking a centralized platform, teams may have to set up "throw away" assembly lines on a project-by-project basis. This causes repeating and overlapping costs related to platform installation, toolchain maintenance, and software licences.

The overall cost is a combination of expenditures in three categories:

Direct costs - costs from actual tooling related maintenance and support work e.g. hours spent installing, maintaining, servicing, problem solving, and supporting the toolchain. Project specific assembly lines with overlapping licences may cause additional cost compared to a centralized platform model.

Indirect costs - costs accumulating as lost working hours while the platform is not in use due to e.g. poor platform maintenance and/or insufficient incident management practices.

Opportunity costs - time spent by development, quality assurance, or operations experts in servicing the toolchain instead of focusing on their primary assignment.

As a rule of thumb, neglecting the work that generates direct costs will cause hard-to-measure but often more costly indirect expenses.

Direct costs

Hours spent in:

- Tool installation
- Expanding the platform with new tools and tool extensions (also testing the extensions prior to assigning them to production)
- Continuous maintenance: keeping track of the updates, update installation, system fine-tuning, capacity planning, etc.
- Providing tool related user support and helpdesk service
- Tool integration and maintenance of the existing integrations
- Building and keeping up the support/maintenance competence
- Developing the tool set: planning, design, installation of new functionalities
- Maintaining the underlying HW/OS platform
- Building and keeping up the HW/OS platform maintenance competence
- Incident management and recovery (including practice rounds)
- Tool licence management and optimization

Indirect costs

Hours lost due to:

- Problems in tool availability
- Problems in tool-to-tool integrations
- Insufficient tool update procedures (e.g. lost efficiency due to the use of old tool versions)
- Insufficient or slow user support

Opportunity costs

Time that should have been spent on delivering value to customers (new features, better quality...) is wasted on basic tool maintenance and support. Experts cannot focus on their area:

- Development work
- Testing work
- Operations work
- DevOps methodology development

Overall cost estimates by calculating the "DevOps tax"

What is the DevOps tax?

Automation provides visible benefits for any software producing organization. The build, deployment and test processes become faster and more reliable, which improves quality and increases value throughput.

However, these benefits do not come free of charge. An increased level of automation equates to an increased number of automation-related tools and integrations. The maintenance burden required to keep the systems up and running increases with the number of tools and integrations.

People involved in the process end up spending part of their valuable time in tool maintenance. This time spent in toolchain maintenance, time that could have been used for more value-adding R&D work, is called the “DevOps tax”.

Estimates with a 10% DevOps tax

According to Forrester research, “developers spend on average 10% of their time in tool maintenance”. By using the 10% rule as a starting point, it is relatively easy to estimate the amount of DevOps tax paid, in terms of person months.

A team of 10 developers loses 1 person year worth of time in tool maintenance that could have been spent on more effective value adding development. This means that on average, at any given time, 1 full time engineer’s worth of effort is reserved for tool related maintenance and support tasks.

Outsourcing some (let’s assume a moderate 50%) of the toolchain maintenance to a service provider for a team of 20 developers has the same effect as hiring a new full time employee. The benefits over hiring a completely new person are obvious - there’s no need to go through time consuming recruitment and onboarding processes because “new employee” is already known by the organization. The “new” employee already knows the work details and is ready to contribute from day one.

Direct cost factors in more detail

Tool maintenance and development work estimates

This chapter gives further guidance for calculating direct tool maintenance and development efforts. The actual work hours spent in each of the tasks depends on multiple factors:

- Skills and experience of the maintenance and support personnel
- Level of maintenance automation
- Toolchain scale and complexity - no. of tools, tool extensions, and integrations
- Amount of toolchain data payload
- Number of (active) users
- Environment type (on-prem, public cloud, etc.)
- Expected system uptime and support
- SLA targets

Relative work loads - tool platform wide tasks

The tasks described in the following table apply to the complete toolchain as a whole. The effort estimates are relative and based on practical experience. For example, a task graded at 12 takes 12 times the effort of a task graded at 1.

Task	Relative effort (units per month or year)
Network management (VPN, authentication, firewalls etc.)	1
System change management	6
Tool vendor management (licences, etc.)	1
24/7 incident management, SLA based	6
Disaster recovery dry runs	3
Kernel security updates (inc. preparations and testing)	3
Backup testing	3
Monitoring, analyzing and reacting to monitor data	12
New feature tracking and analysis	6
Personnel training and competence development	12

Relative work loads - tool specific tasks

The following tasks are tool specific, which means that the complete workload of a toolchain can be calculated by multiplying the tool specific work effort by the number of tools in use.

Task	Relative effort (units per month or year)
Troubleshooting	24
Database migrations	1
Environment management and scaling (servers)	24
Environment management and scaling (cloud)	12
Tool performance tuning	3
Staging environment data updates (inc. preparations and testing)	3
Tool specific security fixes	6
Tool version upgrades	18
System development (planning, design, installation of new functionalities)	12
Personnel training and competence development	12
Supporting the rest of the organization in tool configuration and usage	24

Risks related to insufficient toolchain

Neglecting to properly maintain the software assembly line carries significant risks:

- Savings in direct maintenance will always generate additional indirect costs because the toolchain will not work reliably, causing downtime.
- Unreliable systems lower production throughput and cause overall dissatisfaction within the R&D teams.
- Failing to keep the system up to date via frequent and systematic upgrades will generate technical debt that accrues over time. Paying back the debt after a prolonged period of poor maintenance can be difficult and very expensive.
- A poorly managed system is a source of severe security related risks.

Benefits of a professionally managed toolchain

The benefits of a professionally managed software assembly line are clear:

- The system works reliably and efficiently, enabling the teams to focus on more value adding work and deliver with higher throughput.
- The software R&D development, quality and operations experts are happy because they do not have to spend a significant part of their time in tool maintenance.
- Systematic, gradual toolchain development ensures that it develops in line with needs and requirements and doesn't accrue unnecessary technical debt.
- The systems, including critical assets like requirements, documentation, source code and binaries stay secure.

Automation brings enormous savings and benefits, but we need to be aware that there are costs too. Learning how to measure and mitigate them is essential if you want to avoid accumulating technical debt and remain truly agile. To find out how you can reduce the “DevOps tax” in your company talk to one of our experts.

Contact us:

eficoderoot@eficode.com

www.eficoderoot.com

eficoderoot@eficode.com

🇫🇮 +358 207 40 11 22

🇩🇰 +45 31 68 98 75

🇳🇴 +47 48 67 63 60

🇸🇪 +46 76 340 30 50

🇩🇪 +49 172 4 15 16 17

🇳🇱 +31 20 280 41 18

eficode

www.eficode.com