Software Improvement Group

2020 Benchmark Report

Through the SIG looking glass

Contents

A Letter from Dr. Luc Brandts and Dr. Magiel Bruntink

As we enter a new decade and manage through the impacts of a global pandemic, one thing is certain: the value of digital channels and operations is now more important than ever. "Digital-first" business strategies will continue to power organizations, underscoring the importance of software as a critical success factor.

Software Improvement Group (SIG) has been at the forefront of getting software right since its inception in 2000. For the past 11 years, we've been building our software assurance benchmark database, which now contains more than 36 billion lines of code in more than 280 languages, from more than 5,000 assessments of software systems. This data provides us with unique insights into the state of software assurance and key industry trends.

We published our first report on the state of the software industry in 2019, presenting the most significant takeaways from our benchmark data at the first annual SIG Symposium in Amsterdam.

We're now proud to share with you the 2020 Benchmark Report. Like last year's report, this edition includes a number of interesting insights:

Firstly, we take a look at the general state of affairs. Is build quality still improving, as we saw last year? Or has the trend changed? What are the driving factors behind the trend? Then, answering a question we often receive, we examine the differences between industries.

Who is performing best, and who has seen the greatest improvement? What's the reason behind this? Similarly, we show the differences between the key tech stacks in use. Where do we see the strongest performance?

Next, we explore the impact of software quality monitoring. Last year, we completed a first-time analysis showing a positive impact of SIG monitoring, which prompted us to take a deeper dive. This year's report delivers remarkable new insights on this important topic.

Finally, in these unprecedented times, it would be impossible to ignore the massive changes resulting from the coronavirus crisis. Working from home has become the new standard, to name just one. And we were curious, as we're sure you are, too, as to what the impact of this new arrangement has been on the production as well as build quality of software. This report will tell you.

It has always been our mission to create a healthier digital world. We believe the results and valuable insights revealed in this report can be applied in your daily work and hope they will help you to further improve the digital health of your own organization.

Sincerely,

Dr. Luc Brandts
CEO

Dr. Magiel Bruntink
Head of Research

SIG Software Assurance by the Numbers

470+
customers

280+
technologies from mainframe to mobile

36+ billion
lines of code in software assurance database

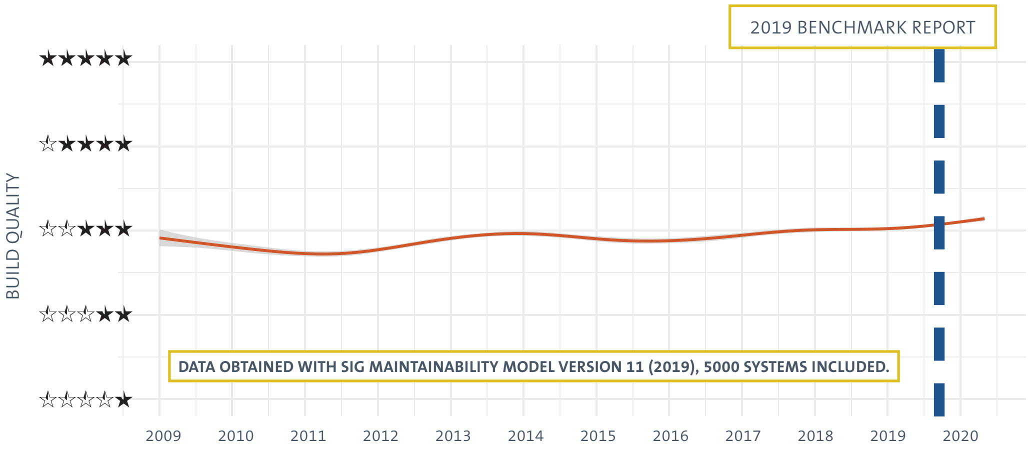
190,000+
system snapshots

170+ million
lines of code analyzed weekly

5.070+
systems evaluated

The Software Build Quality Trend

Last year's Benchmark Report revealed that software build quality has steadily, but very slowly, increased over the last ten years. We're pleased to see that this trend has continued.



[graph 1]

This analysis was done using the 2019 model, applying it to all years, for consistency. Next year, we will be using the 2020 model; so as the industry average per definition is 3 stars, next year's line will be (slightly) lower. Note that this also means that organizations will have to improve their software to stay at the same level. If you're at industry average today, but take no action toward improving your code quality, you will find yourself below par next year.

The drivers behind this continued trend can be found in the details behind this analysis. We're seeing that code is, again, generally smaller and less complex than before. This trend is strengthened by the growing usage of Low Code and BPM solutions, but this is certainly not the only reason. We're also observing this trend in other tech stacks, such as Microsoft .NET-based technologies. One warning sign is that Component Independence is trending down year-over-year. With components generally becoming smaller, there also tends to be more of them, making it increasingly difficult to maintain a proper architecture. At this stage, this negative element of more complex architectures is still outweighed by the other factors. As the trend continues, however, this negative effect could potentially reverse it. At some point, this trend will counter the overall positive trend, so care should be taken. We are keeping a close eye on architectural developments, also by introducing new components to our architectural measurement model.

Software Build Quality
Ranking by Industry

A popular question for debate is whether performance between industries can be evaluated and compared. Is it true that software made in the government sector is really the worst out there, or is this a perception created by the media? Wouldn't the software business itself be the top performer, as the true specialists in the world?

Please note that these are industry averages based on many thousands of systems. In each industry, we find both strong and weak performers. Here, we present the average, so caution should be taken in drawing broad conclusions.

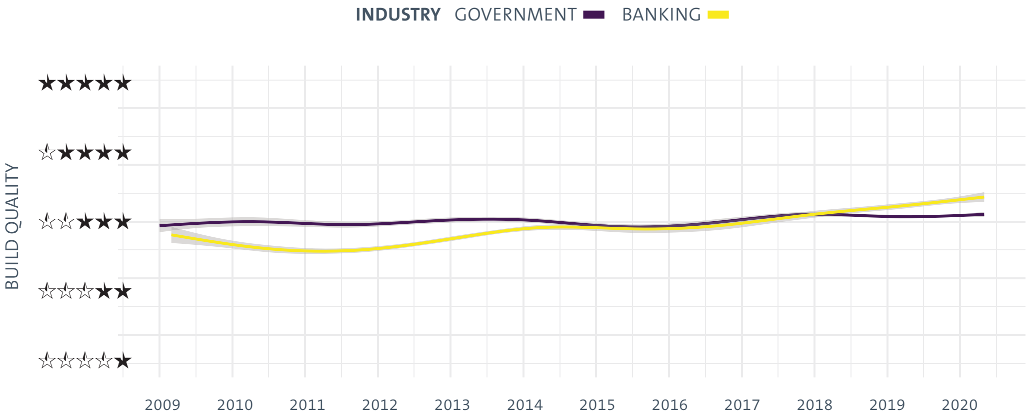
Rank	Industry	Average rating in 2017 - 2020
1	Energy, Oil & Gas	3,29
2	Industrial Transportation	3,27
3	Financial Services	3,23
4	Banking	3,15
5	Government	3,03
6	Insurance	3,03
7	Media	2,90
8	Health Care	2,85
9	Retail	2,81
10	Support Services	2,78
11	Software & Computer Services	2,72
12	Telecommunications	2,65

We can first conclude that the results are quite revealing, and the differences are significant. The difference between the best-performing industry, Energy, Oil and Gas, and the worst-performing industry, Telecommunications, is more than a half star rating. Telecommunications, an industry of great significance, places dead last in 12th, well below the industry average.

Of course, many more industries exist. For some, we simply don't have enough data to provide a valid ranking. If you'd like to learn more about how your industry is performing, please contact us at the e-mail address at the end of this report.

When looking at Government, we first see that this sector ranks just above average. Given the inherent complexity of government software, this is not a poor result. Government has some unique aspects much different from other industries. Requirements often stem from laws and regulations resulting from compromise and long debate – and not necessarily aligned with IT possibilities. In addition, as opposed to most other sectors, Government typically needs to serve a very broad community, for which all exceptions need to be considered. Prior legislation must also be kept intact, leading to inherently more complex systems. It is quite noteworthy that Government ranks ahead of the Software & Computer Services industry. Here, especially, it is important to note that there are many outliers in both directions. That being said, there is still a lot of ground to cover. SIG develops at a minimum of 4 stars as an acceptance criterium, and we advise other organizations to strive for the same level.

There are many more interesting details hidden in these analyses. For example, we compared the quality of software between the Banking and Government sectors over the last ten years. There, we see that for approximately the first seven years, Government was actually performing (slightly) better than Banking. However, in the last three years, Banking has taken a tremendous leap in quality, clearly overtaking Government. Apparently, the banking world has paid extra attention to software quality, which demonstrates very strong steps forward.



[graph 2]

Software Build Quality
Ranking by Tech Stack

Similarly, we analyzed the differences between the various tech stacks. As we conduct assessments on more than 280 different technologies, some of them quite obscure, we've bundled them into six groups:

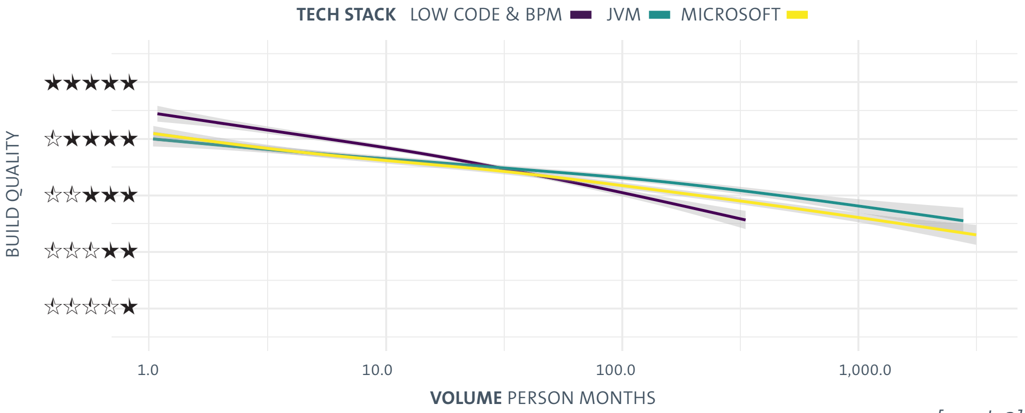
- JVM, Java-related technologies
- Microsoft .NET-related technologies
- Packaged solutions, customizations of configurable ERP, CRM systems
- Legacy technologies, such as COBOL, and other 3GL/4GL languages
- Scripting and mobile technologies
- Low-code and BPM technologies

Rank	Technology stack	Average rating in 2017 - 2020
1	Low code & BPM	3,23
2	Scripting and Mobile	3,21
3	JVM	3,20
4	Microsoft	2,93
5	Packaged solutions	2,57
6	Legacy/3GL/4GL	2,36

Perhaps not surprisingly, the Legacy category yields the poorest results in build quality. Obviously, this is due to the fact that these technologies are less advanced, but it's also the result of them typically being part of highly-complex systems. The fact that they're still out there is, in part, due to their complexity that has, thus far, prevented their replacement. We also see that customizations of packaged solutions are still considerably below industry average; despite a gradual increase in quality over the past years, it's clear that the industry needs to improve.

The top three technology stacks are very close, within hundredths of stars, and hence too close to declare a clear winner. Systems in Microsoft technologies are trailing a bit, leading to a below-average score. Please note that this difference wouldn't grant the preferred selection of Java over Microsoft; for that, the differences are still too small, and there would be too many other factors influencing such a decision.

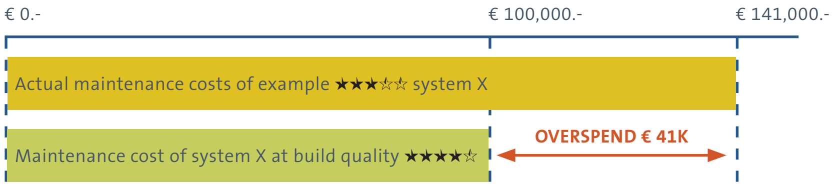
Low Code & BPM technologies are modern software development technologies that facilitate high build quality, given that they're applied in the right way. For example, our analysis shows that for larger systems, Java and Microsoft-based technologies are on par and even better than Low Code & BPM. From that, we can derive that Low Code & BPM technologies should primarily be used for what they're best designed for: smaller applications.



[graph 3]

The Impacts of Software Monitoring

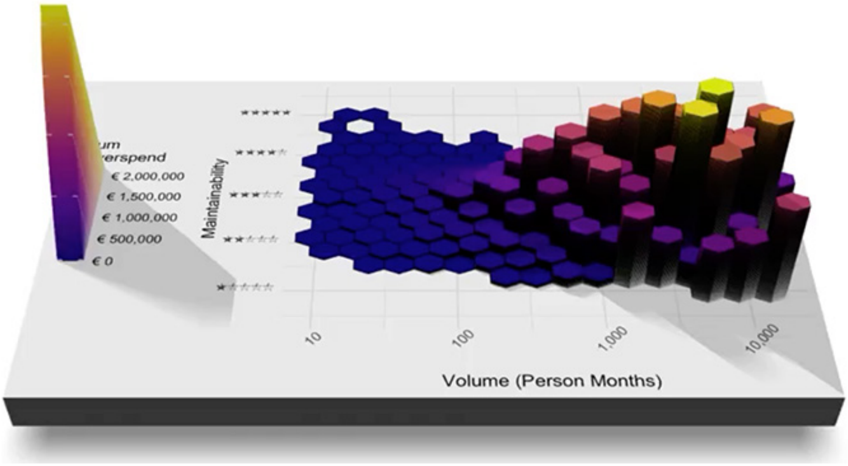
The next topic is near and dear to our hearts, as it pertains to the relevance of our services. The higher the quality of software, the lower the Total Cost of Ownership (TCO). Based on our research, we have been able to estimate this cost of ownership. When looking at the thousands of systems we monitor, from very small to very large, we're able to put them into perspective and show the overspend. The overspend of TCO is calculated based on the idea that, ideally, a system would be built at 4 stars.



[graph 4]

The picture above illustrates the concept of overspend. Take a system that is built at a 3-star build quality level with annual maintenance costs of 141,000 Euro. If that same system would have been built at a 4-star build quality level, we estimate a maintenance cost of 100,000 Euro annually. This implies an overspend of 41,000 Euro for this 3-star system every year.

Now, let's take the concept one step further. For all the systems in our software analysis database, we can now calculate this overspend. Systems



[graph 5]

with lower build quality will have more overspend; larger systems will be more expensive to maintain, so their overspend will be higher as well. When we add up all the overspend of all the systems, we end up with the “Legacy Mountain” shown in the 3D plot on this page.

Next, we look at the impact of SIG monitoring on build quality. Does it make a difference? Spoiler alert: Yes, it does. And quite a bit, actually.

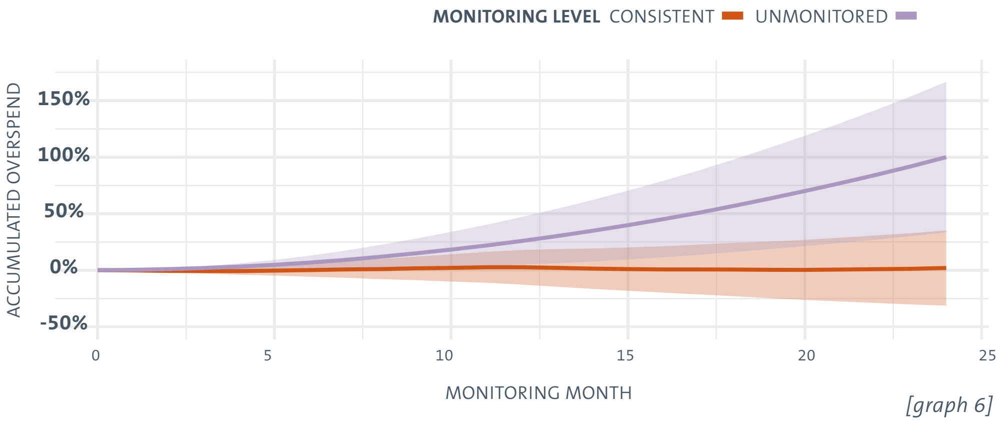
We differentiate between systems that are monitored:

- consistently
- inconsistently
- not at all

The third group requires some explanation. In order to evaluate build quality level, the system obviously needs to be measured. But how could we know the quality of systems we aren’t actively monitoring? This third group is composed of open source systems, so we actually do measure them, unbeknownst to the developers. In order to calibrate our benchmark model, SIG measures a large group of open source systems with developers around the world. These systems are selected in collaboration with TÜViT, a globally-recognized evaluation and certification organization, which also serves as the independent auditor of the SIG software measurement models. This test group can be used as a reference for systems who develop over the course of their lifetime without actively steering on quality, other than employing standard methods. The differences are staggering and clearly demonstrate the benefit of continuous SIG monitoring.

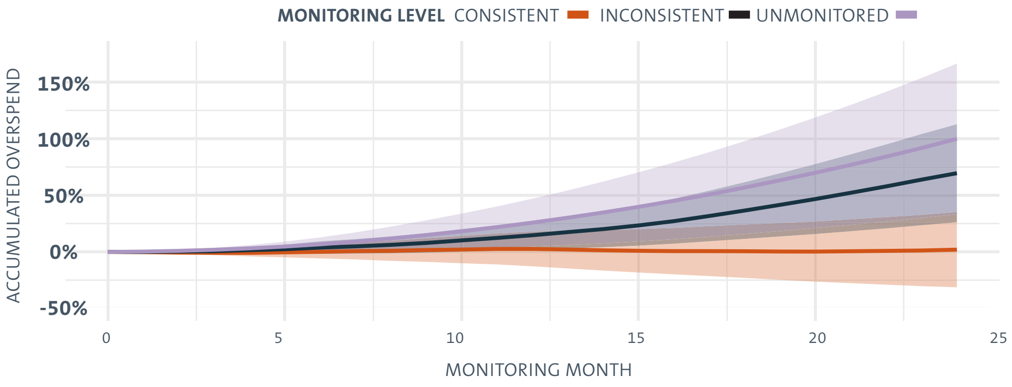
The following graphs show how systems accumulate overspend during 24 months of development. The overspend has been normalized to the unmonitored group; it therefore shows 100% there in month 24.

The first graph [graph 6] clearly shows that the accumulation of overspend in unmonitored systems increases at a rapid pace, leading to the Legacy Mountain pictured on the previous page.



[graph 6]

The next graph [graph 7] includes systems monitored inconsistently. Here, we see that inconsistent monitoring has a positive effect, but significantly less than consistent monitoring. Consistent monitoring helps to keep the feedback cycle as short as possible.



[graph 7]

These graphs show that SIG monitoring has a direct impact on the quality of the software, and, as a result, on the maintenance overspend as well. It shows that even in the worst case, systems monitored by SIG perform better than the best performing unmonitored systems. For larger systems, the overspend can go into the many hundreds of thousands of Euros per year. And the impact on speed of development and, thus, innovation can't be forgotten. As shown in research conducted by SIG back in 2011[1], making changes to a 4-star system can be done three and a half to four times faster than in a 2-star system. Let that sink in: modifications can be made almost four times faster in a system of high build quality. And when comparing a 5-star system to a 1-star system, the differences are even more staggering: development in a 5-star system is more than ten times faster than in a 1-star system. Intuitively, most people grasp the significance of these findings, but probably still underestimate the true magnitude of the impact. At SIG, we have the numbers to support that intuition.

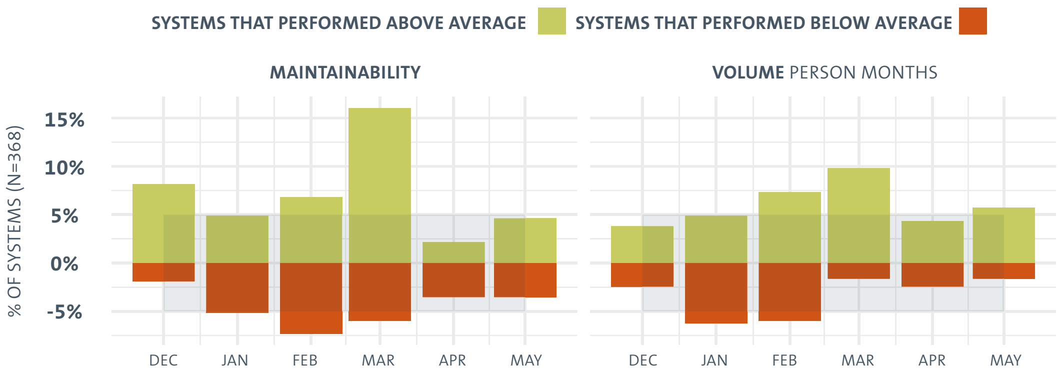
The overspend is the financial factor, significant enough in and of itself, but the development speed factor may be an even more compelling reason to continuously clean up the mess. The analysis shows that consistent SIG monitoring helps maintain a high build quality – and with that, a low cost of ownership and high speed of innovation.

[1] *Bijlsma, D., Ferreira, M.A., Luijten, B., & Visser, J. (2011). Faster issue resolution with higher technical quality of software. Software Quality Journal, 20, 265-285.*

The Impacts of COVID-19 on Software Build Quality & Productivity

Without any doubt, the coronavirus has had the largest impact on our society in recent memory. One of the immediate effects of the pandemic has been that many people have begun working from home. Anecdotally, we've heard stories about teams whose production has skyrocketed, as well as those whose production has slightly decreased.

The SIG development team has seen a rather constant level of production with no real impact from remote working. But these are all anecdotes, not a fact-based trend. As SIG is in the unique position of analyzing thousands of systems regularly, we're able to provide this insight. Has COVID-19 and the resultant governmental measures had an impact on the production of code? And, at the same time, has the quality suffered? Or has it even benefited?



[graph 8]

We're comparing data from the months of March, April and May, 2020 (pandemic-influenced) with data prior to March, 2020 (not pandemic-influenced).

We've built a statistical model to predict the expected changes in build quality and production of code, based on long-term historical data. With this model, we're able to predict what to expect in terms of both elements within a band of normal monthly variation. Without going into the statistical details, the above graph shows that something interesting happened in March. With previous months being more or less within the expected band, March appears to be a clear outlier. We see a remarkable peak in systems that increased their build quality above expectation. Where we normally would have seen a maximum of 5% of systems showing exceptional changes, this month shows more than 15%. Similarly, we notice that 10% of systems showed production above expectation, while just a small percentage fell below. However, things seem to return back to normal in April and May.

The conclusion could be that COVID-19 hasn't, thus far, had any impact on build quality, nor production. But we suspect there's more to the story; clearly, something positive happened in March. Did procedures change? Did ways of collaboration become less formal? Was there greater opportunity for improvisation by teams? Whatever it is, it had a positive effect. We encourage development teams to think about the positives we had in March that seem to have been lost again in April and May.

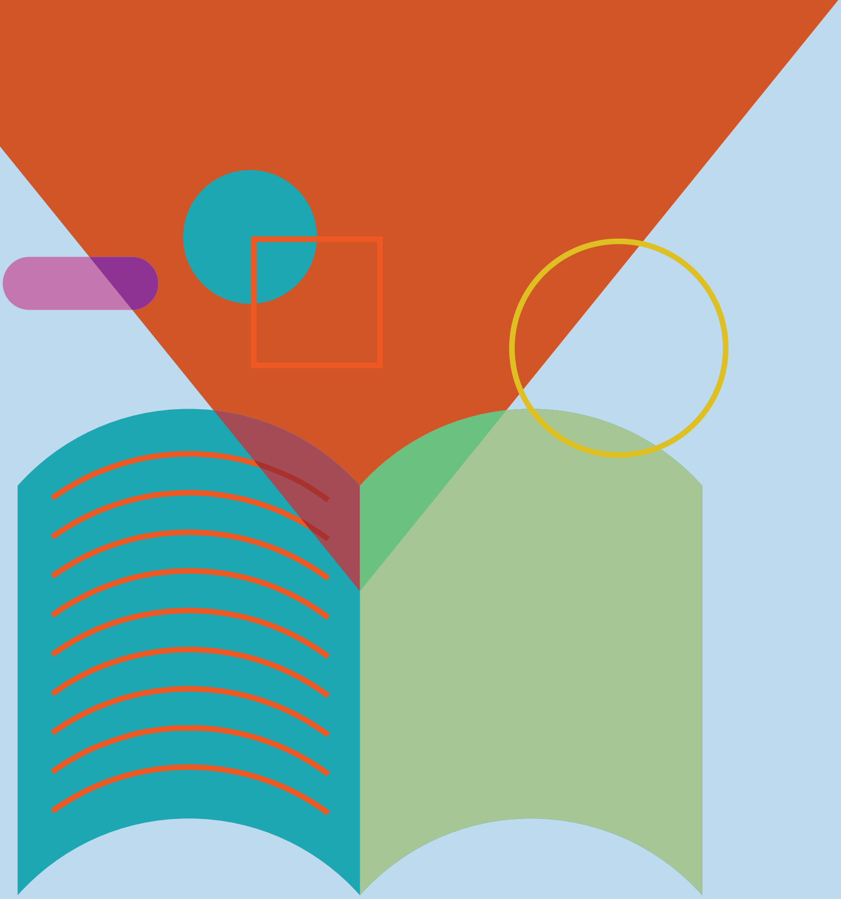
At SIG, such events and observations make us curious. Stay tuned for further updates while we continue this research.

Final Thoughts

We're pleased to see that the build quality of software continues to gradually improve. However, at the same time, there are underlying trends that have the potential to reverse this, which leads us to remain cautious. The positive impact of Low Code may be outdone by a lack of focus on architecture and quality in general, especially for larger systems.

It's also encouraging to see what the impact of focus can be. The progress in quality demonstrated by the Banking industry is a clear example of the significant step forward that can be made with true focus. As this is an analysis based on quite a large number of systems, the results actually speak volumes. The analysis also shows, however, that there's still a lot of ground to cover.

That being said, perhaps the most important conclusion of this analysis was the impact of monitoring. Our analysis demonstrated that software build quality gradually decreases when it's not being consistently monitored; just as a garden needs constant care, software requires attention in order to make the necessary progress. A garden with a lawn that's mowed only occasionally will never be prize-winning. Software is no different.



Colophon

Copyright © 2020
by Software Improvement Group (SIG)
Authors: Dr. Luc Brandts and Dr. Magiel Bruntink

The information in this document may not be copied or published, distributed or reproduced in any way whatsoever without the prior written consent of SIG or the legal consent of the owner.

Design: Plushommes
Art: Rawpixel

Software
Improvement
Group

Fred. Roeskestraat 115
1076 EE Amsterdam
The Netherlands

www.softwareimprovementgroup.com
marketing@softwareimprovementgroup.com

Getting software right for a healthier digital world