wipro.com

SDLC: Debug the cost

Enterprise Business Integration

Testing as a practice has made immense strides. With the rise of DevOps and Continuous Delivery methodologies, the luxury of a separate test window has evaporated. Development and testing have become near-simultaneous activities. Testing has also grown in sophistication. It now includes automation, test data and environment management and is witnessing the emergence of quality engineering where testers have development skills and developers have testing skills. The focus of these testing methodologies has been to improve the productivity of testers and catch bugs faster. The faster a bug is caught, the shorter is the time to market. This is undoubtedly good. But nevertheless this leads to re-work, because the bugs already exist. Experience has shown that the cost of rework is over 50% in most large projects. Can this re-work be brought down? And if it can, where do you start in the Software Development Lifecycle (SDLC)?

Experience has shown that the cost of rework is over 50% in most large projects. Can this re-work be brought down?

The real cost of poor requirements capture

The answer lies in preventing the bugs from getting introduced into code. Drill this down one more level and we see that a significant way to reduce introducing bugs is to improve the requirements capture process. Among the top problems leading to bugs is poor requirement capture. Experience says that common natural language is easily the most pervasive way to capture requirements. A very small percentage of requirements are written in structured natural language or in a formalized language. Using common natural language leads to ambiguity, miscommunication and frequent changes that generate bugs in the development phase. One of the key reasons for this state of affairs is that developers never get to meet clients. They look at requirement documents created by analysts. Ultimately, the outcome can be at two extremes, both equally dangerous: One, the product can pass testing with flying colors, but can turn out to be a poor product because it was developed and tested using wrong or inaccurate requirements; two, it may fail testing completely, throwing up scores of bugs. In both instances, rework will eat into budgets and time to market.

Here is an example of ambiguity from a requirements document, “Shut off the pump if the water level remains above 100 meters for more than 4 seconds.”[1] From this requirement statement it is not clear if the water level refers to mean/median/root mean square/minimum. The interpretation of the requirement is a function of the reader’s background.

Much of the problems in requirements capture have become acceptable norms in development. But what if the requirements can be articulated in a non-ambiguous manner to reduce the number of bugs being introduced in development? In other words, what if we could automate requirements capture?

New tools are becoming available that run through text (written and spoken) and figure out ambiguities. Once these are flagged by the tool, they can be taken back to the client and put into more structured language. These tools surface ambiguities and they can do it at scale and speed – both of which are of critical importance in today’s competitive environment.

1. Detecting Ambiguities in Requirements Documents Using Inspections, Parnas, Asmis and Madey: https://pdfs.semanticscholar.org/fc2e/bbdbdbc21cff575a8dd511bbd8a14574335f.pdf

Automatic code generation: the new challenge

We have the tools that automate requirements gathering. It is the next step of coding that needs equal attention. Code automation has not been fully developed. Usable models, templates, tools, libraries in common/target languages, etc., that can automate development are still viewed with caution. But some methodologies, when used intelligently, can make developers more productive and reduce the risk of bugs. The challenge here is to figure out the automation tools that work and those that don't in specific instances and environments.

In integration projects, the process of creating a spreadsheet mapping spec (done by a business analyst) and then converting that into code (by a developer) amounts to major duplication of effort. Typically, over 50% of interfaces developed in an integration process are simple in nature. In such instances, the simple interface receives an XML (eXtensible Markup Language) file that is run through an XSLT (Extensible Stylesheet Language Transformations) which transforms the input XML according to the logic defined in the mapping spec. What if the XSLT and other output files could be created using the input and output XSDs (XML Schema Definition)? The code that will be generated as a result is then run through an automated testing framework. We would then have an automated assembly line for SDLC starting from requirements gathering right up to the deployment of tested code to the production environments.

Such an approach will not always result in zero errors. But over a period of time, using Machine Learning, the system will know enough so as to be almost 100% accurate.

Thanks to faster computers and new insights using Artificial Intelligence (AI), it is becoming possible to create tools that write code. The goal should be to use those that automatically generate clean, non-verbose and maintainable code.

There is no risk involved in trying these methods. However, the real risk lies in not using these methods: Organizations that do not, will continue to pay the price of re-work and lost market opportunities.

About the author

Gourisankar Mukherjee
Practice Manager, Wipro

Gouri helps clients and partners solve business challenges by leveraging technology enablers and solutions. He has over 20 years of experience in a vast array of technologies including middleware, core Java, Android and Open Source stacks. He is particularly keen on exploring new technologies and loves to get hands-on with them whenever the opportunity presents itself. He can be reached at gourisankar.mukherjee@wipro.com.

Wipro Limited
Doddakannelli, Sarjapur Road,
Bangalore-560 035,
India

Tel: +91 (80) 2844 0011
Fax: +91 (80) 2844 0256
wipro.com

Wipro Limited (NYSE: WIT, BSE: 507685, NSE: WIPRO) is a leading global information technology, consulting and business process services company. We harness the power of cognitive computing, hyper-automation, robotics, cloud, analytics and emerging technologies to help our clients adapt to the digital world and make them successful. A company recognized globally for its comprehensive portfolio of services, strong commitment to sustainability and good corporate citizenship, we have a dedicated workforce of over 170,000, serving clients across six continents. Together, we discover ideas and connect the dots to build a better and a bold new future.

For more information, please write to us at **info@wipro.com**

IND/BRD/MAY 2017-APR 2018