DOI 10.54596/2958-0048-2024-4-213-225
UDK 338.4
IRSTI 81.81.17

BRIDGING THE GAP BETWEEN THEORY AND PRACTICE IN SOFTWARE QA

Kulikova V.P.[1*], Kulikov V.P.[1], Kulikova E.V.[2]
[1]Manash Kozybayev North Kazakhstan University NPLC*
Petropavlovsk, Kazakhstan
[2]Smart Solution, located Aurora, Ontario, Canada
[*]Corresponding author: valentina@ku.edu.kz

Abstract

The article explores the gap between academic training in software testing and the realities of working in the industry. The results of hypothesis testing are presented in the form of a conversation. Using a dialogue between a student, a professor, and a senior QA specialist as an example, key challenges faced by graduates in transitioning from academic settings to real-world professional activities are discussed. The professor explains that the university's software testing course is based on systematic principles, covering core testing methodologies and tools. Meanwhile, the experienced QA specialist provides practical examples, emphasizing the importance of adaptability in dynamic work settings, where project requirements often shift in terms of time, budget, and scope. The article focuses on how theory and practice in software testing can complement each other to achieve optimal results, even with limited resources.

Keywords: Academic training; Software testing; Industry realities; Adaptability; Theory and practice.

ТЕОРИЯ МЕН ПРАКТИКАНЫҢ АРАСЫНДАҒЫ АЛШАҚТЫҚТЫ ЖОЮ: БАҒДАРЛАМАЛЫҚ ҚАМТАМАСЫЗ ЕТУ САПАСЫН ҚАМТАМАСЫЗ ЕТУ САЛАСЫНДАҒЫ

Куликова В.П.[1*], Куликов В.П.[1], Куликова Е.В.[2]
[1] «Манаш Қозыбаев атындағы Солтүстік Қазақстан университеті» КеАҚ*
Петропавл, Қазақстан
[2] Smart Solution, Аврора., Онтарио, Канада
[*]Хат-хабар үшін автор: valentina@ku.edu.kz

Андатпа

Мақалада тестілеу саласындағы теориялық дайындық пен индустриядағы практикалық жұмыс шарттары арасындағы алшақтық қарастырылады. Гипотезаларды тексеру нәтижелері сұхбат түрінде ұсынылған. Студент, профессор және QA аға маманы арасындағы диалог мысалында түлектердің оқу аудиториясынан кәсіби ортаға ауысу кезіндегі негізгі қиындықтары талқыланады. Профессор университеттің бағдарламалық қамтамасыз етуді тестілеу курсы жүйелі принциптерге негізделгенін, тестілеудің негізгі әдістемелері мен құралдарын қамтитынын түсіндіреді. Сонымен қатар, тәжірибелі QA маманы уақыт, бюджет және жобаның ауқымы жиі өзгеретін динамикалық жұмыс жағдайларында икемділіктің маңыздылығын көрсететін практикалық мысалдармен бөліседі. Мақалада шектеулі ресурстар жағдайында теория мен практиканың бағдарламалық қамтамасыз етуді тестілеуде бір-бірін қалай толықтырып, оңтайлы нәтижелерге қол жеткізуге болатынына баса назар аударылады.

Кілт сөздер: Академиялық дайындық; Бағдарламалық қамтамасыз етуді тестілеу; Саланың шынайы жағдайлары; Икемділік; Теория мен практика.

ПРЕОДОЛЕНИЕ РАЗРЫВА МЕЖДУ ТЕОРИЕЙ И ПРАКТИКОЙ В ОБЛАСТИ ОБЕСПЕЧЕНИЯ КАЧЕСТВА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Куликова В.П.[1*], Куликов В.П.[1], Куликова Е.В.[2]

[1]НАО «Северо-Казахстанский университет имени Манаша Козыбаева»*
Петропавловск, Казахстан
Smart Solution, Аврора, Онтарио, Канада
[*]Автор для корреспонденции: valentina@ku.edu.kz

Аннотация

В статье рассматривается разрыв между теоретической подготовкой в области тестирования и практическими условиями работы в индустрии. Результаты проверки гипотез представлены в формате интервью. На примере диалога между студентом, профессором и старшим специалистом QA обсуждаются ключевые вызовы, с которыми сталкиваются выпускники при переходе от учебных аудиторий к реальной профессиональной деятельности. Профессор объясняет, как курс по тестированию программного обеспечения в университете строится на системных принципах и охватывает основные методологии и инструменты тестирования. Тем временем, опытный специалист QA делится примерами из практики, подчеркивая важность умения адаптироваться к динамичным условиям работы, где постоянно меняются требования по времени, бюджету и масштабу проектов. Основное внимание уделено тому, как теория и практика тестирования могут работать вместе для достижения наилучших результатов, несмотря на ограничения в ресурсах.

Ключевые слова: Академическое обучение; Тестирование ПО; Реалии отрасли; Адаптивность; Теория и практика.

INTRODUCTION

Software testing and quality assurance (QA) are the unsung heroes of the development process. They ensure that the end product isn't just functional but reliable, secure, and efficient. While theoretical knowledge about QA can be acquired through courses, the real challenge comes in applying this theory to real-world projects where constraints like time, cost, and scope constantly shift.

At North Kazakhstan University, the "Testing and Quality Assurance of Software" course provides a solid foundation in QA methodologies and tools. However, how well does this theory prepare students for the dynamic and high-pressure environment of modern software development? Through a conversation between a student, a professor, and a senior QA professional, this article explores the gap between academic preparation and real-world application.

The article presents in the form of an interview the partial results of a large-scale study, including conclusions obtained using expert assessments (Condorcet, Borda, Kemeny median methods), as well as A/B testing using ANOVA analysis for parametric and non-parametric comparison. This structured and iterative approach ensures a comprehensive analysis that integrates theoretical considerations with empirical testing to address the research objectives effectively.

THEORETICAL BASIS. METHODOLOGY

The research methodology comprises the following stages, all conducted iteratively to refine hypotheses and improve the analysis at each step:

1. *Collection and Identification of Alternatives (Indicators):*

- Conducting interviews and/or brainstorming sessions to identify relevant indicators.
- Output: A comprehensive list of potentially relevant indicators [1, 2, 3, 4].

2. *Reduction of Alternatives:*

- Reducing the dimensionality of the list using the Borda method to prioritize indicators [1].

- The final dimensionality of the list is determined based on the context.

3. *Ranking and Consolidation of Alternatives:*

- Applying the Kemeny method to identify the consensus alternative [4].

- Using the Condorcet method to construct an ordered ranked list [1, 5].

4. *Selection of Metrics and Hypothesis Formalization:*

- Defining metrics that describe the indicators [6, 7, 8, 9, 10].

- Formalizing hypotheses for empirical testing [2, 11, 12, 13, 14].

5. *A/B Hypothesis Testing* (Context-Specific Application [11, 15, 16]):

- Testing the null hypothesis of no significant differences between results in two samples:

✓ *Parametric* tests: Student's t-test, Fisher's F-test for independent populations, G-test of mean differences [2, 7, 17, 18].

✓ *Nonparametric* tests: Fisher's randomization test, Wilcoxon two-sample test, Mann-Whitney U-test, binomial test for large samples (contingency table analysis 2×2), median test [2, 19, 20].

- Analysis of Variance (ANOVA) to evaluate the effect of the studied factor levels:

✓ *Parametric* methods: Analysis of Variance (ANOVA), Bartlett's test, Cochran's G-test, Scheffé's test [2, 15].

✓ *Nonparametric* methods: Kruskal-Wallis test, median test, Jonckheere-Terpstra test, contingency table analysis (k×r) [2, 19, 20].

- Factor relationship analysis:

✓ *Parametric* methods: Pearson's correlation coefficient, covariance, multiple correlation [11, 21].

✓ *Nonparametric* methods: Spearman's correlation coefficient, Kendall's tau, concordance coefficient, Jaccard similarity index, Hamming metric [16].

6. Substantive Interpretation of Results:

- Conducting a qualitative analysis of expert assessments and statistical testing outcomes [13, 14, 16].

- Developing recommendations based on statistically significant results [8].

The following methods were most frequently used:

- *The Condorcet, Borda methods, and Kemeny median* were selected due to the stability of the expert group's core despite variability in the composition of experts depending on the context of each hypothesis. These methods account for disagreements and variability in opinions, ensuring the correct aggregation of evaluations even when the group composition changes. The Condorcet and Borda methods are suitable for processing ranked data, which is crucial in the presence of diverse opinions and hypothesis contexts. The Kemeny median is used to determine the "average" preference among multiple possible sequences, helping identify the most common preferences among experts and "smooth out" disagreements.

- *A/B testing (including ANOVA analysis)* complements expert assessments by providing objective quantitative measurements of the effects of different conditions. However, not all data in the study met the assumptions for parametric tests (e.g., normal data distribution or quantitative scale). In such cases, non-parametric tests were used as a more flexible alternative for data analysis, as they do not require strict adherence to distribution or measurement scale.

The Kemeny median minimizes the total distance between a set of expert opinions in order to obtain a median consensus opinion that deviates the least from the opinions of all experts.

Let x_{ik} – represent the opinion of the i-th expert on the k-th issue, expressed as a rank or score assigned by the i-th expert to k-th issue. The distance r_{ij} between the opinions of the i-th and j-th experts on each k-th issue is measured as $r_{ij}^k = |x_{ik} - x_{jk}|$. The total distance between the opinions of the i-th and j-th experts across all issues is calculated using the Manhattan norm as follows: $r_{ij} = \sum_k |x_{ik} - x_{jk}|$. The total disagreement of the i-th expert's opinion with all other experts (i.e., disagreement with the group) is computed as: $R_i = \sum_j r_{ij} = \sum_j \sum_k |x_{ik} - x_{jk}|$. The consensus opinion x^* is the opinion of the expert that exhibits the least disagreement with the group, i.e. $x^* = \arg\min_i R_i$.

The Borda Method. Assume that each expert has an individual preference order for a set of alternatives A. If an expert prefers an alternative x∈A over an alternative y∈A, this is denoted as x>y. The individual preference order of an expert represents a permutation of the alternatives arranged in order of preference. The collection of all individual preferences of the experts is referred to as the preference profile of the experts. For a given preference profile, each alternative is assigned points by an expert as follows: the alternative ranked last receives zero points, the alternative ranked second-to-last receives one point, and so on. The alternative ranked first receives (n−1) points from the expert, where n is the total number of alternatives. The best alternative according to the Borda method is the one that accumulates the highest total score from all experts.

The Condorcet Method. Despite the possibility of a paradox, the Condorcet method remains widely recognized. The algorithm proceeds through the following steps:

1. Initialization. Let A = {a1, a2,…,an} be a set of alternatives, and let there be a group of experts, each with a preference order over all alternatives.
2. Pairwise Comparison. It is assumed that all alternatives are distinct. In practice, if alternatives are identical, they can be merged or one of them can be excluded, as pairwise comparisons of identical alternatives do not affect the outcome. For each pair of alternatives (ai, aj), where i≠j, ai∈A, aj∈A:
   - ✓ Count the number of experts who prefer ai over aj (denoted N(ai>aj)).
   - ✓ Count the number of experts who prefer aj over ai (denoted N(aj>ai)).
3. Determination of Pairwise Wins. For each pair (ai, aj)(a_i, a_j)(ai,aj):
   - ✓ If N(ai>aj)>N(aj>ai) alternative ai wins over aj.
   - ✓ If N(aj>ai)>N(ai>aj) alternative aj wins over ai.
4. Checking for a Condorcet Winner. If there exists an alternative ak∈A that wins against all other alternatives in pairwise comparisons, ak is considered the Condorcet winner.
5. Condorcet Paradox. If no alternative wins against all others (i.e., a preference cycle arises, such as ai>aj>ak>ai), there is no Condorcet winner, and the algorithm cannot unambiguously determine the most preferred alternative.

A/B Testing for Comparing Two Independent Samples with Identical Distributions (or Equal Characteristics, e.g., Means):

For instance, a *parametric* Student's *t*-test with a formal null hypothesis (comparison metric: population means) $H_0 = \bar{x}_1 = \bar{x}_2$, and a two-tailed alternative $H_1 = \bar{x}_1 \neq \bar{x}_2$. Depending on whether the variances are equal or unequal, the test statistic is given as t~N(0;1), where N(0;1) – denotes the standard normal distribution. At the predefined significance level α, the alternative hypothesis H_1 is accepted if the calculated *t*-statistic exceeds the critical value t_α determined from the N(0;1) distribution table.

ANOVA Analysis.

For the *parametric* test (analysis of variance), the data table consists of x_{ij}, where $i = \overline{1,k}$ – are the levels of factor A, $j = \overline{1,n_i}$ – the observation number for each level. The actual Fisher criterion statistic F is calculated. Upon selecting a significance level α, we find the critical value Fα from the F-distribution table such that, for degrees of freedom k_1= k-1 and k_2=n-k, the following condition holds: $P\{1/F_{\alpha;k1;k2} > F \text{ or } F > F_{\alpha;k1;k2}\}=\alpha$

If the observed value of F exceeds $F_{\alpha;k1;k2}$, the null hypothesis is rejected, meaning the factor has an effect on the result. Otherwise, the null hypothesis is not rejected, and it is considered valid; i.e., at the significance level α, the sample means are statistically equal, and the factor's effect is not significant.

For the *nonparametric Kruskal-Wallis test* (with arbitrary alternatives), the data table consists of r_{ij}, where $i = \overline{1,k}$ – denotes the levels of factor A, and $j = \overline{1,n_i}$ – denotes the observation number within each level. The calculation of the test statistic for the null hypothesis test, which assumes the equality of treatment effects on samples (with more than two samples) with unknown but equal means, is as follows:

$$H = \frac{12}{n(n+1)}\sum_{i=1}^{k}\frac{R_i^2}{n_i} - 3(n+1)$$

where R_i, $i = \overline{1,k}$ – the sum of the ranks for the i-th group, k – is the number of groups, n_i – the size of the i-th group, and $n = \sum_{i=1}^{k} n_i$ – is the total number of observations. Under the null hypothesis, the distribution of the statistic H with (k-1) degrees of freedom follows the chi-squared distribution: $H \sim \chi^2_{k-1}$. With a correction for tied ranks, the statistic is adjusted as: $H_{adj} = H - \sum t_i^2/n$, where t_i – represents the number of tied ranks (observations) within the i-th group.

Calculation Tools: MS Excel [17, 26], SPSS [22, 23], and Python [24, 25] were used for all statistical calculations, ensuring reliable and replicable results.

INTERPRETATION OF RESEARCH RESULTS FRAGMENTS

Introducing the Actors

The core of the research group consists of 2 SKU professors, employees "Smart solution": BA manager/solution architect (Eve) – 1 person, QA manager/Release manager – 1 person, Senior QA analyst/automation – 1 person, QA – 2 persons, QA analyst – 2 persons, Junior QA – 1 person, Senior BA – 2 persons, BA – 4 persons. Variable part employees "SRB": Release manager – 1 person, QA manager – 1 person, QA lead – 1 person, Senior QA – 1 person, automation specialist – 2 person, QA – 6 person, BA manager – 1 person, Senior BA – 1 person, BA – 2 person.

In the course of a discussion between the participants – a student, a professor, and an experienced QA specialist – the results of the research are synthesized. Key findings, problematic issues, and potential IT solutions are discussed, enabling the alignment of theoretical aspects of QA with the practical realities of the industry.

Student: A curious and motivated student at North Kazakhstan University, taking the "Testing and Quality Assurance of Software" course. His goal is to understand how the theory of QA aligns with actual industry practices.

Professor: A seasoned professor at North Kazakhstan University. He emphasizes the systemic nature of QA, showing how it connects with other disciplines and how QA professionals fit into the broader software development lifecycle.

Eve: Eva is a Senior Business Analyst and Solution Architect at Smart Solution (Canada), a software development company for banking systems. Her practical experience and university education in "Information Technology in Economics, Business and Management" have helped her excel in a variety of roles. Eve's work bridges the worlds of QA, business analysis, and solution architecture, allowing her to offer valuable insights into the day-to-day realities of QA in a fast-paced development environment.

Example of Methodology Application Results

1. Based on expert interviews and surveys, a list of priority indicators for the research on "Indicators for the successful integration of an IS graduate into the role of a tester" was formed: {a) Theoretical knowledge and its adaptation to practice, b) Flexibility and adaptability in conditions of limited information, c) System complexity and business significance analysis, d) Automation process management, e) Work with functional and non-functional testing, f) Ability to leverage the benefits of "shift-left" and continuous testing, g) Impact of time constraints, h) Collaboration and communication, i) Specialization and new technologies, ...}

2. Using the Borda ranking method [1], the list was narrowed down to {a), b), d), f), h), i)}.

3. A consensus-based "average" opinion of the experts on the key skills and indicators for successful integration into work was identified [9], based on hiring experience and releasing projects: the priorities are {b), a), d), f), h), i)}.

4. Key metrics of the study [6, 7, 8]:

- Compliance with theoretical software quality standards: Share of test cases that comply with ISO/IEC 25010 standards (%); Coverage level of theoretical requirements in test documentation (%).
- Testing and quality assurance efficiency: Average number of detected defects per 1000 lines of code (Defect Density); Average time to detect and fix defects (Defect Detection Efficiency); Share of repeating defects (%).
- Analysis of the level of practical application of theoretical approaches: Percentage of testing methodologies used (e.g. TDD, BDD) in real projects (%); Level of test automation (share of automated tests among all tests, %);
- Comparison of theoretical expectations and practical results: Difference between predicted and actual quality metrics (e.g. performance, reliability, functionality); Average execution time of test cases compared to theoretical estimates.
- Quality and completeness of test documentation: Level of compliance of checklists and test cases with theoretical recommendations (%); Requirements Coverage Ratio.
- Education and training of specialists: Level of education: university, college, courses; Percentage of employees certified in theoretical approaches (e.g. ISTQB); Level of employee competencies assessed through testing of knowledge of theory and practical skills.

5. A/B testing was used to compare independent groups. The conclusions from A/B testing were drawn with a confidence level not lower than 90%.

Understanding the Basics: From Theory to Practice

Student: "Professor, can you explain the key principles of testing and QA that we're learning in this course?"

Professor: "Certainly. The 'Testing and Quality Assurance of Software' course at North Kazakhstan University is built on a systemic approach to QA. It's not just about learning how

to test software, but understanding how QA integrates with other subjects like software engineering, project management, and even statistics. For example, the lifecycle of software is one of the main pillars we cover, and we look at it through three models: Waterfall, Agile, and DevOps. Each model assigns different roles and responsibilities to QA.

We teach students about various testing methodologies, such as functional testing, non-functional testing, usability testing, performance testing, and more. Automation is also a core part of the course. Students learn which parts of QA can be automated, and we introduce tools like Selenium, JUnit, and LoadRunner. Documentation is another area we cover extensively—what a QA professional should deliver in terms of test plans, bug reports, and even project estimates.

But QA is more than just testing against specifications. In our course, we also focus on critical thinking and systemic problem-solving. QA professionals must understand the business impact of their work, how software interacts with other systems, and even how data and statistics come into play during quality assurance."

Eve: "From my experience, having a solid foundation in different testing methodologies is crucial, especially as you begin to encounter more complex software systems. For instance, in my current role at Smart Solution, where we develop banking software, we don't just check if the system works—we evaluate how a simple transaction on the front-end affects back-end processes like general ledger entries, reporting, and customer notifications.

But there's a big difference between theory and practice. One of the challenges you'll face in the real world is balancing thoroughness with project constraints. The 80/20 principle applies perfectly here. The principle, originally an economic concept, suggests that 80% of the effects come from 20% of the causes. In QA, 80% of bugs can often be found with just 20% of the testing effort, while the remaining 20% of bugs may take 80% of your time to uncover and fix. It's not about perfection; it's about knowing when to stop. One QA analyst I worked with was incredibly thorough, sometimes too thorough. He would dive deep into areas outside the scope of the ticket, such as looking at back-end reports and GL entries when the bug was just on a front-end form. While his diligence uncovered valuable insights, it sometimes delayed the current task unnecessarily. You need to know when to stop, balance quality against cost, and prioritize what's important."

Student: "We've been learning about different testing methodologies like functional and non-functional testing. In real-world projects, how do you decide which methodology to use, or are they often used together?"

Professor: "In theory, each methodology serves a distinct purpose. Functional testing ensures that the software works as intended, while non-functional testing focuses on how the system performs under certain conditions, such as speed or security. In real-world applications, these methodologies are often used together. For example, you may start with functional testing to ensure that core functionalities work as expected, then move into non-functional testing to assess how the system handles stress or scales under load. The key is knowing how to prioritize based on the project's needs and time constraints."

Eve: "In most real-world projects, you'll often use a combination of methodologies depending on the context. Functional testing is essential to ensure that the software works as intended, but non-functional testing, like performance testing, is just as critical, especially for large systems like the ones we work on in banking.

For example, we have reports in our banking system and programs that update the database, such as posting scheduled transactions or bill payments. Every time we make changes related to these processes—whether it's adding fields to the bills framework or redesigning

reports — QA doesn't just test the new field or layout. We also conduct performance testing to make sure that these changes don't negatively impact the system's overall performance. This is critical because even small changes can slow down transaction processing, which can have significant consequences in a banking environment.

So, while functional and non-functional testing are distinct, they are often used together to ensure both correctness and efficiency. In practice, knowing when and how much of each type of testing to do is about balancing priorities, deadlines, and the potential risk to the system."

Tools and Methodologies: The Essentials and Real-World Application

Student: "Professor, what about the tools we've been learning? Are these the same tools used in the industry, or should we expect to learn new tools once we start working?"

Professor: "Good question. In our course, we focus on foundational tools like Selenium for automated testing, JUnit for unit testing, and LoadRunner for performance testing. These tools are widely used across many industries, and learning how to use them gives students a strong starting point. We also emphasize that the specific tools you use will often depend on the company's needs, project requirements, and even team expertise.

Beyond the tools themselves, we teach methodologies. Whether you're working in a Waterfall, Agile, or DevOps environment, the tools may vary, but the principles of testing remain consistent. Understanding these principles will help you adapt to any tool in the future."

Eve: "The tools you learn at university are a great foundation, but don't get too attached to them. The landscape of QA tools changes quickly. In the real world, you might be working with a combination of new tools and legacy systems. For example, in Smart Solution, we often customize tools or scripts to meet specific project needs, and this can differ from project to project.

And remember, the best tools in the world won't help if your environment doesn't support them. Sometimes, practical constraints, like shared environments, mean you must compromise. Ideally, QA teams should have separate environments from developers to ensure clean testing. But in practice, especially in smaller teams or fast-moving projects, you might have to share. When that happens, it's essential to have protocols in place. Everyone needs to clean up after themselves, notify the team of any changes, and keep the process simple—if the protocol is too convoluted, people won't follow it."

Student: "In your experience, how quickly do new tools or technologies become standard in the industry? Should I focus on mastering a few, or try to stay up to date with the latest tools?"

Professor: "This is a great question. Technology evolves rapidly, and while it's important to stay updated, you should focus on mastering core tools that have withstood the test of time, like Selenium for automation or JUnit for unit testing. These tools give you a solid foundation and are widely used across industries. However, don't stop there. Keep an eye on emerging tools, but don't spread yourself too thin by trying to master everything. Your ability to adapt and quickly learn new tools when required is what will set you apart."

Eve: "That's spot-on. Tools change, but the principles behind testing remain consistent. In my experience, at my previous company, there was a heavy emphasis on formal test plans that outlined not only the tools but also the approach, scope, resources, roles, and timelines. For example, a typical test plan included sections like:

- ✓ Test Objectives
- ✓ Scope of Testing
- ✓ Testing Resources
- ✓ Test Environment
- ✓ Risks and Contingencies

✓ Exit Criteria

We also had a very rigid release cycle – every six months. But at my current company, things are more agile. We don't spend much time on formalities; the QA team only documents test cases to serve as proof during audits. The focus is on getting the work done, ensuring that there are no critical errors, and moving fast with a release cycle every two months instead of every six. It's a much quicker pace, which means we focus on what's essential and don't get bogged down in over-documentation."

Challenges in QA: Theory vs. Reality

Student: "I sometimes get stuck on test cases during my assignments. Is this normal, and how should I handle such situations?"

Professor: "Challenges are part of the learning process. It's normal to feel overwhelmed, especially when transitioning from theory to practice. My advice is to break problems down into smaller, manageable parts. A good QA professional not only understands how to write test cases but also how to analyze requirements and figure out what the software is supposed to do. Testing isn't just about finding bugs; it's about understanding the intent behind the code and verifying whether it meets the business's needs."

Eve: "In the real world, testing often begins with incomplete or high-level requirements. You'll rarely get the luxury of having everything neatly defined. This is something I had to learn the hard way. No university course really prepared me for giving accurate time estimates in such conditions. When BAs provide only high-level requirements, the company might not want to spend resources on detailed analysis unless they're certain the client will pay for it. As a QA, you're often expected to come up with estimates based on limited information.

My advice: learn to ask the right questions. If you don't have enough details, flag it and ask for more. But remember, there's always pressure to deliver. You need to strike a balance between getting enough information to test effectively and moving the project forward."

Student: How do time constraints impact the development process, particularly when it comes to QA?

Professor: According to theory and industry best practices, time constraints should be managed through careful project planning and prioritization. In an ideal scenario, all tasks, including thorough testing, are allotted sufficient time to ensure quality. Timeboxing techniques and setting clear milestones allow teams to focus on the most critical aspects first, ensuring that essential components are completed on time. When under pressure, teams may need to prioritize high-risk areas and use risk-based testing to allocate resources effectively. However, cutting corners is always risky, especially in industries where quality is tied to legal or financial consequences, such as banking.

Time constraints can create a significant impact on QA processes. QA teams often face the pressure of ensuring that all major areas are tested while keeping up with deadlines. Risk-based testing, where the highest priority is given to the most critical functionalities, is key in such situations. The theory says that by focusing on high-risk areas first, the QA process can still protect against major failures even when the timeline is compressed.

Eve: While perfect testing is ideal, in reality, we must always weigh the trade-off between exhaustive testing and delivering on time. What matters most is focusing on the highest risk areas to maintain both quality and client trust.

Nobody likes cutting corners, especially when real people's money is involved. In our case, if something goes wrong due to an error on our side, the penalties can be steep—our company would have to pay compensation to our clients.

Recently, we worked on a huge project where we needed to update our system to support a 360-day year for interest accruals, which is a requirement in the Philippines. Our system was set up for 365 and 366-day years, so this new requirement meant a major change. Interest calculations are one of the most critical and central parts of any core banking system because they define the income and expenses for credit unions and banks. They also have legal implications for customers.

Here's the tricky part: the code change itself involved just two lines, in two places. But that tiny change required half the system to be retested! The QA team had to retest everything from data entry, to how the values were displayed, to report generation, and even transaction posting.

Of course, we couldn't spend six months just on retesting, so the QA team worked closely with the developers and BAs. They figured out which parts of the system used the shared code and identified critical parts versus those that weren't used as often. We had to see if there were any safe corners to cut to deliver the project on time. This is where risk-based testing really comes into play—identifying what's mission-critical versus what we can afford to test less thoroughly. But even then, we had to be very careful. After all, the last thing we want is to miss something important when dealing with financial data

Student: Can you give an example of one of the challenges your company faced in real life?

Eve: You know, planning is essential for any company. My company has a long-term roadmap, spanning three years, and a short-term plan with yearly release cycles—we have six of them per year. We try our best to schedule projects throughout the year, aligning them with those six releases. The scheduling takes into account factors like complexity, client demand, our internal product improvements, overall company goals, resource availability, and so on. It's not easy, given all the parameters involved.

But here's the challenge: we develop banking software, which is highly regulated. Regulators like the CRA (Canada's Tax Agency), FSRA, Bank of Canada, Bank of Jamaica, and others often issue new regulations that our clients—credit unions and smaller banks—must comply with. When that happens, it can completely disrupt our carefully planned schedule. While we try to anticipate these possibilities, they can still catch us off guard.

For example, we had planned to release a significant enhancement to our mobile app, allowing non-Canadian financial institutions (FIs) to top-up mobile phones through the app, which integrates with our core banking system. However, in September, new must-have regulations for income reporting were announced for both Canada and Jamaica. These regulations needed to be implemented before the end of the year, forcing us to adjust our plans and shift the release of the mobile app feature to accommodate the regulatory changes.

This kind of scenario is common in our field—balancing planned innovations with the need to stay compliant with sudden, mandatory updates.

The Future of Testing and QA: What's next?

Student: "What can we expect in the future of QA? What trends should we be paying attention to?"

Professor: "The future of software testing and QA is rapidly evolving, driven by technological advancements, increased automation, and the demand for more complex software solutions. Here are some key trends and innovations that will shape the landscape in the coming years:

1. Increased Automation with AI and Machine Learning:

Automation is already playing a huge role in software testing, but the introduction of AI-driven tools will revolutionize how we conduct tests. AI can help predict which areas of the software are most likely to have bugs based on historical data. Machine learning algorithms can also optimize regression testing by identifying which test cases are most crucial to execute. However, automation doesn't mean testers will become obsolete – human oversight will always be necessary to ensure that the tools are working as intended.

2. Shift-Left and Continuous Testing:

Testing is moving earlier into the software development lifecycle, a trend known as 'shift-left' testing. Traditionally, QA teams were only involved after development was completed, but now we see QA integrated into every phase, from requirements gathering to deployment. Continuous testing in Agile and DevOps methodologies ensures that testing is a constant process, providing quicker feedback and allowing teams to detect and resolve issues early.

3. Quality Assurance Beyond Testing:

As you've learned in the course, QA isn't just about finding bugs—it's about ensuring the overall quality of the product, including performance, usability, and compliance. In the future, QA teams will likely take on more responsibility in terms of ensuring that software meets all regulatory and industry-specific standards, especially in fields like healthcare, finance, and AI ethics. QA professionals will need to stay informed about changes in these industries to guide teams through compliance and risk management.

4. Collaboration with Developers and Product Teams:

Cross-functional teams are becoming the norm, with QA analysts working closely with developers, business analysts, and product managers. In this environment, soft skills like communication, negotiation, and project management will be just as important as technical skills. Testers need to collaborate on building not just functional software, but also software that meets the strategic goals of the business.

5. Specialization in Emerging Areas:

With the rise of fields like blockchain, IoT (Internet of Things), and quantum computing, QA roles will also need to evolve. Testers will have to specialize in these areas to understand their unique challenges. For example, blockchain requires testing for both security and performance, as well as ensuring that smart contracts work as intended. IoT introduces the challenge of testing distributed networks of devices that need to operate reliably in real-time. Quantum computing, though still in its infancy, will require testers who can handle completely different architectures and paradigms of computation."

Eve: "In my experience, automation will continue to expand, especially in areas like regression testing. But don't forget, there are still parts of QA that require a human touch. Usability testing, for example, relies heavily on how real users interact with the software. Also, as cybersecurity becomes more critical, security testing will play an even bigger role in the future.

But a word of caution: while automation is great, it's not a silver bullet. Automating everything can lead to inefficiency. You need to weigh the cost of setting up automated tests against the value they bring. The 80/20 rule applies here as well – sometimes, 20% of your automated tests will catch 80% of the bugs. Focus on automating repetitive, high-value tasks, but don't over-automate just because you can."

Conclusion: Bridging Theory and Practice in QA

Professor: "In summary, we aim to equip you with a broad understanding of QA principles, tools, and methodologies. However, as Eve highlighted, balancing the thoroughness

of testing with project constraints is something that only comes with experience. You'll need to find that balance in your future roles, learning how to prioritize and optimize your approach based on the context of the project."

Eve: "Exactly. In reflecting on my journey from academia to the industry, I can confidently say that while university taught me the fundamentals, it was the day-to-day challenges in real projects that honed my skills. Learning when to adapt, how to ask the right questions, and recognizing when 'good enough' is truly enough—these lessons come with experience and are invaluable in the fast-paced world of software development."

In the end, while technologies may change, the fundamental skills of problem-solving, communication, and a deep understanding of the software lifecycle will always be essential to a successful QA career.

Professor: "In conclusion, preparing students for a career in QA requires a comprehensive understanding of principles, tools, and methodologies, but the role's evolving nature also demands adaptability. As Eve emphasized, specialization and the adoption of new technologies—such as AI, machine learning, and DevOps—are becoming indispensable. Balancing theoretical rigor with practical application ensures our graduates are ready to meet industry challenges while remaining open to continuous learning."

Eve: "Absolutely. From my own transition from academia to the industry, I've learned that while theoretical foundations are crucial, the practical aspects—like understanding CI/CD pipelines, using AI for automation, and navigating the fast pace of real projects—are where true growth happens. It's essential to adapt quickly, ask critical questions, and embrace 'good enough' solutions in the right contexts, especially with shifting priorities and constraints."

Final Reflection

Despite the rapid development of technologies, core competencies such as critical thinking, collaboration, and understanding the software development lifecycle remain foundational to a successful QA career. The balance between theoretical knowledge and practical experience is not just an advantage, it is a necessity in this dynamic field.

Let us accept as an axiom: the expanding role of quality assurance specialists requires mastering new technologies and methodologies, including AI-based automation, CI/CD processes, and DevOps principles. A successful career in QA requires:

✓ *Fundamental knowledge of CI/CD and DevOps practices* to effectively integrate testing into the development and quality maintenance process at all stages of the product lifecycle.

✓ *Skills in using AI for test automation and data analysis*, enabling significant improvements in testing efficiency and accuracy, as well as the ability to promptly identify potential issues.

✓ *Willingness to learn new tools and technologies*, reflecting the ability to adapt to changes and integrate innovative solutions into daily work.

Thus, the synergy between theoretical knowledge and practical experience, particularly in the context of new technologies, is the key to successfully entering and advancing in the QA profession.

References:

1. Akhrameyko, A.A. (n.d.). *On improving the efficiency of managerial decision-making under conditions of non-stochastic data uncertainty* [Electronic resource]. Retrieved October 5, 2024, from http://bseu.by
2. Kulikova, V.P. (2006). *Data analysis and processing in information systems: Educational and methodological manual*. - Petropavlovsk: SKSU named after M. Kozybayev.

3. Mukhin, O.I. (n.d.). Lecture 36. *Expertise* [Electronic resource]. Retrieved October 5, 2024, from http://stratum.ac.ru
4. RSVPU. (n.d.). *Using Excel to summarize expert opinions using Kemeny's median method* [Electronic resource]. Retrieved October 5, 2024, from http://rsvpu.ru
5. MedStatistic. (n.d.). *Online calculators for statistical indicators* [Electronic resource]. Retrieved October 5, 2024, from http://medstatistic.ru
6. Kulikov, S.V. (2023). *Software testing: Basic course (3rd ed.)*. Moscow.
7. Myers, G.J., Sandler, C., & Badgett, T. (2020). *The art of software testing* (4th ed.). Wiley.
8. Aniche, M. (2022). *Effective software testing*. Manning Publications.
9. Applitools. (n.d.). *Test Automation University* [Electronic resource]. Retrieved October 5, 2024, from https://www.applitools.com
10. Ministry of Testing. (n.d.). *Where software testers, QA, and quality engineers build their careers* [Electronic resource]. Retrieved October 5, 2024, from https://www.ministryoftesting.com
11. StatSoft. (n.d.). *Big Data, data mining, predictive analytics, statistics*. StatSoft Electronic Textbook [Electronic resource]. Retrieved October 5, 2024, from https://archive.org
12. Omsk Region. (n.d.). *Probability theory – Additional resources* [Electronic resource]. Retrieved October 5, 2024, from http://newasp.omskreg.ru/probability/
13. Orlov, A.I. (2018). *Methods of managerial decision-making: Textbook*. Moscow: KNORUS.
14. Borodina, A.V., & Nekrasova, R.S. (2023). *Statistical criteria in data analysis: Educational manual*. Petrozavodsk: PetrSU Publishing.
15. Borovikov, V. (2003). *Statistica (Art of data analysis on a computer)*. Moscow (St. Petersburg): Piter.
16. Habr. (n.d.). *How to choose the right statistical test for different metrics* [Electronic resource]. Retrieved October 5, 2024.
17. Microsoft Support. *Using the analysis toolpak* [Electronic resource]. Retrieved November 5, 2024, from https://support.microsoft.com
18. Nikitin, O.R., & Korneeva, N.N. (2020). *Methods for measuring statistical parameters of radio signals: Textbook*. Vladimir: Vladimir State University named after A.G. and N.G. Stoletovs.
19. BirdyX.ru. (n.d.). *Non-parametric statistical criterion of Kruskal-Wallis* [Electronic resource]. Retrieved November 12, 2024.
20. LibreTexts. (n.d.). *Kruskal–Wallis Test - Statistics* [Electronic resource]. Retrieved November 12, 2024.
21. Crispin, L., & Gregory, J. (2019). *Agile testing condensed: A brief introduction*. AgileTester.
22. IBM. (n.d.). *User guide for IBM SPSS Statistics 27 base system* [Electronic resource]. Retrieved October 5, 2024.
23. SPSS Manual. (n.d.). *SPSS instruction manual* [Electronic resource]. Retrieved October 5, 2024, from http://bsu.ac.ug
24. Real Python. (n.d.). *Python statistics fundamentals: How to describe your data* [Electronic resource]. Retrieved October 5, 2024, from https://realpython.com/python-statistics/
25. Python.org. (n.d.). *Statistics-mathematical statistics functions – Python 3.13.0 documentation* [Electronic resource]. Retrieved October 5, 2024, from https://docs.python.org/3/library/statistics.html
26. Murray, A. (n.d.). *Advanced Excel success: A practical guide to mastering Excel* [Electronic resource]. Retrieved October 5, 2024, from https://doi.org/10.1007/978-1-4842-6467-6

Information about the authors:

Kulikova V.P. – corresponding author, Professor, "Information and Communication Technologies" chair, candidate of technical sciences, associate professor, Kozybayev University, Petropavlovsk, Kazakhstan; e-mail: v4lentina@mail.ru;
Kulikov V.P. – Professor, "Information and Communication Technologies" chair, candidate of physical and mathematical sciences, associate professor, corresponding member of the international informatization academy, Kozybayev University, Petropavlovsk, Kazakhstan; e-mail: qwertyrant@ku.edu.kz;
Kulikova E.V. – Master Science in Computer Science; Solution Architect/Senior Business Analyst/BA team manager, Smart Solution, located Aurora, Ontario, Canada; e-mail: ekulikova@smartsolution.com.