ДИФФЕРЕНЦИАЛЬНЫЙ СОЦИАЛЬНО-ПСИХОЛОГИЧЕСКИЙ АНАЛИЗ ПРОФЕССИОНАЛЬНЫХ ДЕФОРМАЦИЙ В IT-СРЕДЕ

Д.Н.Латипова[1], К.И.Ахметова[2]
[1]студентка университета Renessans
(Узбекистан, Ташкент)
latipova.dilafruz@gmail.com
[2]научный руководитель,
доцент университета Renessans
(Узбекистан, Ташкент)
tdiu_69k@mail.ru
https://doi.org/10.5281/zenodo.21784564

Аннотация. *Статья посвящена дифференциальному социально-психологическому анализу профессиональных деформаций и эмоционального выгорания специалистов в IT-индустрии. На основе триангуляции нейрокогнитивных исследований, качественного анализа дискуссий в профессиональных сообществах (Stack Exchange) и профессиографического метода изучена специфика трёх категорий специалистов: разработчиков ПО (Backend / Frontend / Fullstack), системных и инфраструктурных инженеров (SysAdmin / DevOps / SRE) и инженеров по обеспечению качества (QA/QC). Выявлены ключевые средовые стрессоры, преобладающие когнитивные искажения, механизмы психологической защиты и дифференцированные формы выгорания для каждой группы. Установлено, что для разработчиков ПО основным фактором деформации выступает фрагментация внимания (context switching) и невидимость интеллектуального труда; для системных инженеров - дежурства on-call и антиципаторная тревога; для QA/QC-инженеров - структурная асимметрия ответственности при отсутствии контроля над сроками и снижение бдительности (vigilance decrement). Сформулированы прикладные рекомендации по профилактике выгорания и оптимизации рабочих процессов с учётом специфики ролей.*

Ключевые слова: *профессиональные деформации, эмоциональное выгорание, IT-специалисты, когнитивные искажения, context switching, on-call, QA/QC, DevOps, SRE, хронификация, алекситимия, компенсаторная гиперформализация.*

Abstract. *The article presents a differential socio-psychological analysis of occupational deformations and emotional burnout among IT industry specialists. Based on the triangulation of neurocognitive research, qualitative analysis of professional community discussions (Stack Exchange), and professiographic analysis, the study examines three distinct practitioner categories: software developers (Backend / Frontend / Fullstack),*

system/infrastructure engineers (SysAdmin / DevOps / SRE), and quality assurance engineers (QA/QC). Key environmental stressors, dominant cognitive biases, psychological defense mechanisms, and specific burnout manifestations are identified for each group. Software developers are primarily affected by continuous context switching and the invisibility of intellectual output; infrastructure engineers suffer from on-call duties and chronic anticipatory anxiety; while QA/QC engineers face structural responsibility-control asymmetry and vigilance decrement. Applied recommendations are formulated to mitigate burnout and optimize organizational workflows tailored to each role.

Keywords: *professional deformation, emotional burnout, IT specialists, cognitive biases, context switching, on-call, QA/QC, DevOps, SRE, chronification, alexithymia, compensatory hyperformalization.*

В условиях современной технологической гонки стратегическая устойчивость организаций, разрабатывающих программное обеспечение, всё чаще определяется не пропускной способностью серверов, а психофизиологическими лимитами «человеческого процессора». В индустрии, где объем кодовой базы (LoC) в критических узлах превышает 150 млн строк[1], когнитивная нагрузка на специалиста достигает своего физиологического предела. Статистика «остаточных дефектов» остается пугающе стабильной на протяжении десятилетий: от 1 до 15 ошибок на KLoC[2], что указывает на фундаментальный разрыв между сложностью систем и адаптивными возможностями человеческого мозга.

Современная IT-среда характеризуется формированием особого типа субъекта труда - «индивид-цифровая среда», где программный код и технические средства становятся неотъемлемой частью профессиональной идентичности специалиста. Рассмотрим три типа данных:

- нейрокогнитивные исследования восприятия сложности кода [1], которые демонстрируют, что декодирование кода задействует зоны, ответственные за рабочую память и торможение импульсивных реакций, то есть предъявляет системную нагрузку на исполнительные функции мозга;

- корпус качественных данных из профессиональных сообществ (Stack Exchange) о выгорании и «плохих днях» разработчиков (выгорание определяется как психологическая проблема, характеризующаяся эмоциональным истощением, деперсонализацией и снижением ощущения личной эффективности на рабочем месте);

- профессиографический и стилевой анализ когнитивных паттернов пользователей цифровой среды [3, 5].

Такое триангулирование позволяет развести три уровня анализа:

1. Уровень когнитивной архитектуры деятельности - как мозг специалиста буквально обрабатывает сложность и неопределённость системы;

2. Уровень средовых стрессоров - организационные и технологические паттерны, порождающие хронический дистресс;

[1]McConnell S. C. (1993). Code complete: A practical handbook of software construction.Redmond, WA: Microsoft Press.

[2] Silberg G. (2017). Protecting the fleet and the car business. Amstelveen, NL: KPMG.

Sandu I., Salceanu A., Bejenaru O. (2018). New approach of the customer defects per lines of code metric in automotive SW development applications. J. Phys.1065. 10.1088/1742-6596/1065/5/052006

3. Уровень личностной переработки стресса - защитные механизмы, коммуникативные стратегии и итоговые формы выгорания.

Специфика IT-среды как психосоциального контекста определяется тремя структурными факторами, общими для всех трёх категорий, но по-разному их фрустрирующими: экстремально высокой скоростью технологического обновления (перманентная угроза профессиональной некомпетентности), опосредованностью труда цифровым интерфейсом (дефицит невербальной обратной связи и «эффект чёрного ящика» результата) и культурой перманентной измеримости (метрики, спринты, SLA, инцидент-репорты), формирующей трансформацию восприятия времени и пространства в «профессиограмме» цифрового субъекта [3].

Проведем профильный психологический анализ по категориям. Для разработчиков программного обеспечения (Backend / Frontend / Fullstack developers) несовпадение объективной цикломатической сложности кода и субъективно переживаемой когнитивной нагрузки является основой когнитивных искажений и установок: разработчик систематически недооценивает или переоценивает сложность в зависимости от структуры логических ветвлений, что создаёт почву для двух противоположных искажений - иллюзии контроля («я вижу всю систему целиком») и синдрома самозванца при столкновении с чужим, непрозрачно организованным кодом. Перфекционизм здесь принимает форму эстетизации кода («чистая архитектура» как самоцель), что коррелирует с обсуждениями выгорания на Stack Exchange [2], где пользователи описывают навязчивое переписывание рабочих решений. Гиперответственность проявляется в персонализации багов production-среды как личностного провала, а не системного риска.

Как основной источник стрессоров и деформаций можем рассматривать фрагментацию внимания: непрерывное переключение между контекстом и задачами, пересмотр и анализ написанного кода (код-ревью) является одним из наиболее часто упоминаемых триггеров «плохого дня» наравне с неработающими инструментами и заблокированными зависимостями (blocked dependencies).

Вторичным стрессором является «невидимость» интеллектуального труда: качественная архитектурная работа не имеет немедленного визуального артефакта, в отличие от, например, дизайна, что обесценивает труд в глазах нетехнического менеджмента и порождает хроническую потребность в валидации через код-ревью.

Разумеется, имеются и защитные механизмы, где доминирует интеллектуализация, то есть превращение эмоционального конфликта (например, отвергнутый pull request) в чисто техническую дискуссию о стандартах кода. Часто наблюдается эмоциональное отстранение от продукта («я пишу код, а не решаю проблемы пользователя»), защищающее от тревоги за конечный результат, которым разработчик не управляет полностью. Все это отражается в коммуникации индивида. Типичный паттерн - избегание прямого конфликта через технические аргументы («пассивно-агрессивный комментарий в PR» вместо личного разговора), сложности с делегированием, обусловленные убеждением, что «только я знаю, где спрятана сложность» (форма гиперконтроля, укоренённая в реальном факте фрагментарности знания о legacy-коде).

В случае системных инженеров и инженеров инфраструктуры продуктов (SysAdmin / DevOps / SRE) доминирующим когнитивным искажением представляется катастрофизация ситуаций, то есть профессиональная норма «всё сломается в самый неподходящий момент» рационально обоснована практикой, но при хронизации

превращается в тревожное ожидание катастрофы даже в стабильные периоды (гипербдительность). Гиперответственность здесь максимальна и институционализирована через SLA и error budget - личная метрика (uptime, MTTR) буквально становится численным выражением профессиональной состоятельности.

Ключевой и наиболее специфичный для этой категории профессионалов стрессор - дежурства on-call, разрушающие циркадные ритмы и создающую состояние «условной готовности», при котором даже отсутствие инцидента не даёт полноценного восстановления (антиципаторная тревога).

Второй стрессор, это резкий контраст между рутинной невидимой поддерживающей работой и краткими периодами экстремальной интенсивности при критических инцидентах, что формирует специфический паттерн «дисрегуляции возбуждения», когда организм не успевает адаптироваться между состояниями монотонии и острого стресса.

Для защиты от стрессоров вырабатывается рационализация инцидентов («это была не моя зона ответственности»), которая сочетается с эмоциональным дистанцированием от последствий сбоев для конечных пользователей, это психологически необходимая защита при работе с абстрактными, обезличенными системными метриками. При хронификации возможна алекситимия, тоесть снижение способности идентифицировать собственное эмоциональное состояние вследствие постоянного подавления тревоги ради оперативного реагирования на инцидент.

При коммуникации типичен гиперконтроль (нежелание передавать доступ или процедуры другим членам команды, «bus factor» как результат не только организационной, но и личностной динамики) и лаконичный, инструментальный стиль коммуникации во время инцидентов, который вне контекста может восприниматься как холодность или отстранённость.

Еще одна категория разработчиков - это инженеры по обеспечению качества, или QA/QC-инженеры. Для этой категории в качестве когнитивных искажений специфичен «синдром привратника»: гипертрофированное чувство единоличной ответственности за качество релиза при объективно ограниченном влиянии на процесс разработки. В сочетании с распространённой организационной культурой обвинения QA за пропущенные дефекты («Stop Blaming QA», как отмечается в производных от Obi и др. работах [4]) формируется хронический синдром самозванца иного типа, чем у разработчиков: не «я недостаточно компетентен технически», а «меня считают виновным по умолчанию».

Конечно же, для этой категории ключевым стрессором является структурная асимметрия власти: QA обнаруживает проблему, но не управляет сроками и приоритетами её исправления, что порождает хроническую фрустрацию контроля. Второй стрессор, как было отмечено Джумагуловой и ее соавторами [3] применительно к цифровым субъектам в целом, но особенно выраженный у QA, - рутинизация внимания (повторяющееся тестирование по чек-листам) при требовании сохранять постоянную бдительность к аномалиям - состояние, близкое к «vigilance decrement» в инженерной психологии.

Здесь в качестве защиты часто наблюдается компенсаторная гиперформализация (избыточная документация багов как способ снять с себя ответственность за субъективность оценки) и пассивная агрессия в баг-трекерах - форма выражения фрустрации в отсутствие легитимного канала прямого влияния на приоритизацию.

При коммуникациях наблюдается избегание прямой конфронтации с разработчиками через формализованные каналы (тикеты вместо личного разговора), при

этом внутри QA-команды существует повышенная солидарность и совместное совладание («мы против дедлайнов»), защищающая от индивидуализации вины.

Результаты анализа сведены в единую таблицу и приложены (табл.1).

Таблица 1

Сравнительная матрица категорий

Категория	Ключевой стрессор	Доминирующая защита	Форма выгорания	Уровень риска
Разработчики ПО	Континуальный context switching, «невидимость» интеллектуального труда	Интеллектуализация, эмоциональное отстранение от продукта	Редукция личных достижений (обесценивание собственного вклада)	средний
Системные инженеры (SysAdmin, DevOps, SRE)	On-call дежурства, контраст рутины и критических инцидентов	Рационализация, эмоциональное дистанцирование, риск алекситимии	Эмоциональное истощение (хроническая антиципаторная тревога)	высокий
QA/QC-инженеры	Асимметрия ответственности без контроля над сроками, «vigilance decrement»	Гиперформализация, пассивная агрессия в тикетах	Деперсонализация (циничное отношение к продукту и коллегам)	средний-высокий

Исходя из рассмотренных особенностей можно сформулировать несколько рекомендаций для каждой категории сотрудников.

1. Разработчикам ПО рекомендуется:

– ввести защищённые «deep work»-блоки без прерываний (митингов, пинков в мессенджерах) и учитывать частоту переключений контекста как организационную, а не личную метрику эффективности;

– внедрить практику видимой атрибуции архитектурных решений (документация «почему», а не только «что»), чтобы компенсировать невидимость интеллектуального вклада.

2. Для системных инженеров (SysAdmin, DevOps, SRE) предпочтительно:

– формализовать восстановительное время после дежурства (обязательный отгул/сдвиг графика после ночного инцидента) как политику, а не индивидуальную договорённость;

– распределять экспертизу через регулярные ротации и парную поддержку (снижение bus factor), что одновременно снижает гиперконтроль и институциональный риск.

3. QA/QC-инженерам следует:

– формализовать полномочия QA в процессе приоритизации багов (право голоса в планировании спринта, а не только фиксация дефектов постфактум), устраняя структурную асимметрию ответственности и контроля;

– производить ротацию типов задач между рутинным регрессионным тестированием и исследовательским тестированием для профилактики снижения бдительности (vigilance decrement).

ИСПОЛЬЗОВАННАЯ ЛИТЕРАТУРА

1. Gao Hao, Haytham Hijazi, João Durães, Júlio Medeiros et al. (2022) On the accuracy of code complexity metrics: A neuroscience-based guideline for improvement. DOI: https://doi.org/10.3389/fnins.2022.1065366
2. Antônio Carlos M. de Paula, Carlos Frederico J. Muakad, Sávio Freire, Rodrigo Spínola, Manoel Mendonça. (2024). Burnout in Software Projects: An Analysis of Stack Exchange Discussions. DOI: https://doi.org/10.1145/3701625.3701670
3. А.Ф. Джумагулова и др. Профессиографические особенности деятельности IT-специалистов как рискогенные факторы выгорания. (2024), DOI: https://doi.org/10.22363/2313-1683-2024-21-1-220-241
4. Ike Obi, Jenna Butler, Sankeerti Haniyur, Brian Hassan, Margaret-Anne Storey, Brendan Murphy. (2024). Identifying Factors Contributing to Bad Days for Software Developers: A Mixed Methods Study. (2024), DOI: https://doi.org/10.48550/arXiv.2410.18379
5. И.С. Лучинкина. (2025). Когнитивные особенности личности пользователя цифровой среды: стилевые параметры. DOI: https://doi.org/10.25588/CSPU.2025.187.3.014
6. McConnell S. C. (1993). Code complete: A practical handbook of software construction.Redmond, WA: Microsoft Press.
7. Silberg G. (2017). Protecting the fleet and the car business. Amstelveen, NL: KPMG.
8. Sandu I., Salceanu A., Bejenaru O. (2018). New approach of the customer defects per lines of code metric in automotive SW development applications. J. Phys.1065. 10.1088/1742-6596/1065/5/052006