УДК 004.4-004.9

А.А. ПРИХОЖИЙ, А.М. ЖДАНОВСКИЙ

МЕТОД ОЦЕНКИ КВАЛИФИКАЦИИ И ОПТИМИЗАЦИЯ СОСТАВА ПРОФЕССИОНАЛЬНЫХ ГРУПП ПРОГРАММИСТОВ

Белорусский национальный технический университет

Рассматривается проблема формирования и оптимизации команд программистов с учетом квалификации и уровня владения технологиями и инструментами программирования. Известные технологические среды для организации работы команд, такие как Agile, формулируют лишь общие требования и принципы формирования коллективов и распределения работ между ними. Предлагается метод формализации оценки квалификации отдельных программистов и целых программистских групп. Исходными являются данные о наиболее востребованных технологиях и инструментах программирования, их рейтинге, а также результаты опроса программистов об уровне владения ими. Квалификация группы программистов оценивается с учетом требований к конкретному проекту как величина, интегрирующая три составляющие: среднюю квалификацию программистов, включаемых в группу; квалификацию группы по лучшим представителям по каждой из технологий; пороговые значения уровней квалификации программистов и групп программистов по каждой из технологий, а также пороговые значения интегрированной квалификации, отражающие специфику конкретного проекта. Вклад каждой составляющей определяется варьируемыми. весовыми коэффициентами. Метод положен в основу генетического алгоритма, выполняющего поиск количества, размера и состава программистских групп, которые имеют максимальную суммарную квалификацию. Разработано программное обеспечение и проведены вычислительные эксперименты на выборке программистов, окончивших белорусские университеты. Полученные результаты демонстрируют реальную способность системы находить количество, состав и размеры групп программистов, увеличивающие суммарную квалификации групп до 30% при минимизации числа незадействованных разработчиков. Результаты доказывают практическую значимость разработки в сфере технологий и средств управления профессиональными коллективами программистов.

Ключевые слова: *Программист, технология, инструментарий, квалификация, группа программистов, состав группы, оптимизация.*

Введение

Технология Agile [1] гибкой разработки программного обеспечения позволяет вырабатывать требования и находить решения благодаря совместным усилиям групп разработчиков и заказчиков. Она базируется на адаптивном планировании, эволюционном развитии, постоянном совершенствовании, быстром и гибком реагировании на изменения. Хотя Agile используется во многих средах разработки, она требует дальнейшего развития для распределенных команд программистов. Правильное сочетание экспертов имеет решающее значение при включении разработчиков в команды. Неправильное распределение работ может исключить подключение ведущих специалистов. Поручение работы команде, которой трудно найти подходящего специалиста, увеличивает общие затраты на набор персонала и выполнение работ, затрудняет правильный подбор людей в распределенную команду.

Успех распределенного коллектива программистов в реализации крупного проекта сильно зависит от адекватности используемых технологий и инструментов программирования, а также от умения эффективно декомпозировать проект на части. Эти части должны быть поручены тем группам программистов, которые обладают необходимыми для этого знаниями и технологиями. Проблема оптимального разбиения программистов на группы сформулирована в работе [2]. Она является многофакторной и слабо структурированной. В [2] учтены следующие факторы: производительность каждого программиста, способность пар программистов увеличивать или снижать производительность в процессе совместной работы, возрастание

затрат на интерфейсы между программистами при увеличении числа программистов в группе. На решение подобного рода проблем ориентированы эволюционные методы оптимизации [3, 4]. В работе [5] предложен генетический алгоритм решения проблемы, однако он не учитывает уровень владения программистами технологиями и инструментами программирования, необходимыми для работы над проектом.

В данной статье анализируются современные технологии и инструменты программирования, оценивается уровень владения ими каждым из программистов и целыми программистскими группами, оптимизируется размер и состав программистских групп.

Рейтинг технологий и инструментов программирования

Анализ результатов исследований компании RedMonk [6] о популярности языков и инструментальных средств программирования и результатов исследований организации IEEE Spectrum [7] о рейтинге языков позволил разработать таблицу, описывающую множество T из 16 основных технологий и инструментов. Для каждой технологии указан рейтинг, показывающий значимость и широту применения технологии, а также требование обязательного владения ею хотя бы одним членом группы программистов. По назначению все множество технологий делится на 6 подмножеств. Системы контроля версий и управления проектами включают Git, Tortoise SVN, VJR и TFS с рейтингом 0.3 каждая. К средам разработки относятся Visual Studio и Eclipse, рейтинг обоих 0.6. Системы управления базами данных представлены Oracle SQL (рейтинг 0.5) и Microsoft SQL Server (рейтинг 0.6). Языки программирования представлены Java, C#, Visual Basic, C++, Java Script и XSL с рейтингом 1.0, 0.9, 0.7, 0.9, 0.8 и 0.6 соответственно. Операционные системы представлены Windows и Linux с рейтингом 0.6 и 0.5 соответственно. Следует подчеркнуть, что высокий рейтинг технологии показывает необоснованность создания коллективов программистов, в которых нет ни одного эксперта по данной технологии.

Ключевые технологии и инструменты программирования

№	Название	Код	Рейтинг	Обязать
1	Git	VGT	0.3	нет
2	Tortoise SVN	VTS	0.3	нет
3	TFS	VTF	0.3	нет
4	Jira	VJR	0.3	нет
5	Visual Studio	DVS	0.6	да
6	Eclipse	DEC	0.6	да
7	Oracle SQL	OBM	0.5	нет
8	Microsoft SQL Server	DBM	0.6	нет
9	Java	LJ	1.0	да
10	C#	LC#	0.9	да
11	Visual Basic	LVB	0.7	нет
12	C++	LCP	0.9	да
13	Java script	LJS	0.8	да
14	XSL	LXS	0.6	нет
15	Windows	OSW	0.6	да
16	Linux	OSL	0.5	да

Уровень владения программистом технологией / инструментом

Для решения второй задачи оценки уровня владения программистами технологиями и инструментами использован метод опроса. Каждому из опрашиваемых программистов предложено заполнить анкету, в которой он указывает уровень *level* владения каждой из технологий. Уровень *level* определяется по пятибалльной шкале: 0 – отсутствие знания технологии; 0.25 – минимальное знание; 0.5 – промежуточные навыки; 0.75 – расширенный опыт владения технологией; 1 – знания и опыт эксперта. Экспертом считается программист, который владеет теоретическими экспертными знаниями, разработал не менее двух крупных проектов и проработал не менее двух лет с данной технологией. Программист обладает расширенными навыками, если выполняются два критерия из трех, и обладает промежуточными навыками, если выполняется один из критериев.

Результаты опроса 24 программистов (множество P) с высшим образованием, окончивших белорусские университеты и работающие в программистских фирмах, представлены на рис. 1, где строки соответствуют программистам, столбцы – технологиям из таблицы. На пересечении строки p и столбца t стоит прямоугольник, высота которого указывает на один из пяти уровней 0, 0.25, 0.5, 0.75 и 1 владения программистом p технологией t. Отсутствие прямоугольника означает нулевой уровень. Строки с большей суммарной площадью прямоугольников показывают более квалифицированных программистов. Столбцы с боль-

шей площадью прямоугольников указывают на более востребованные и значимые технологии.

Квалификация программиста

Квалификация программиста $p \in P$ с учетом уровня знания/владения технологиями из множества T оценивается по отношению к максимальному уровню знания/владения выражением

$$Qualif(p) = \sum_{t \in T} rank(t) \cdot factor(p,t) \Big/ MaxQualif \quad (1)$$

где

$$factor(p,t) = \begin{cases} level(p,t), \text{ если } level(p,t) \geq RL_t^p \\ 0, \text{ в противном случае} \end{cases} \quad (2)$$

и

$$MaxQualif = \sum_{t \in T} rank(t) \quad (3)$$

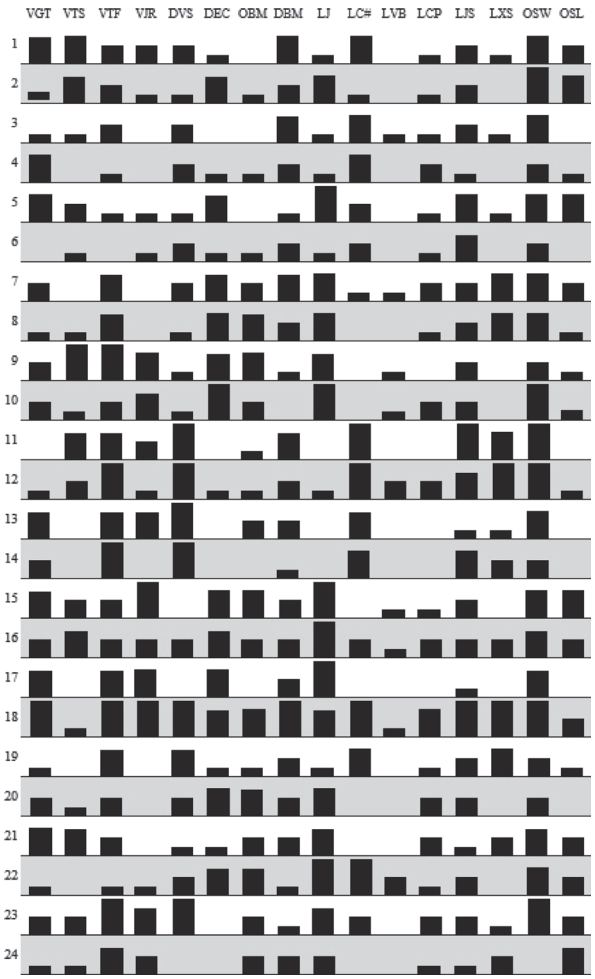


Рис. 1. Уровень 24 программистов владения 16 технологиями

Значение $rank(t)$ из диапазона [0, 1] берется из таблицы. Значение $level(p, t)$ из диапазона [0, 1] определяется результатами опроса, представленными на рис. 1. Пороговое значение RL_t^p уровня $level(p, t)$ позволяет снизить шансы допуска к работе над проектом программистов, обладающих недостаточно высоким уровнем квалификации. Значение выбирается из диапазона [0, 1]. Если $RL_t^p \cong 0$, то к проекту могут подключаться начинающие программисты или программисты, владеющие частью технологий. Если $RL_t^p \cong 1$, то проект ориентирован на разработку экспертами. Как следствие, значение $Qualif(p)$ находится в диапазоне [0, 1]. Согласно формуле (1), программисты, владеющие предпочтительно технологиями с высоким рейтингом, обладают более высокой квалификацией по сравнению с программистами, владеющими предпочтительно технологиями с низким рейтингом.

Рис. 2 показывает квалификацию $Qualif(p)$ каждого из 24 программистов в отношении значимости/владения всеми 16 технологиями с учетом их рейтинга. Значение квалификации находится в диапазоне [0, 1] и является относительным к максимальному значению $MaxQualif = 9{,}5$. Лидером в квалификации является программист с номером 18, за ним идут программисты с номерами 12 и 16.

Квалификация группы программистов

Пусть все множество P программистов разбито на k групп с образованием множества $G = \{g_1, \ldots, g_k\}$. Программисты группы g образуют множество P_g. Средняя квалификация группы g, включающая n_g программистов, определяется как среднее значение квалификаций $Qualif(p)$ по всем программистам $p \in P_g$:

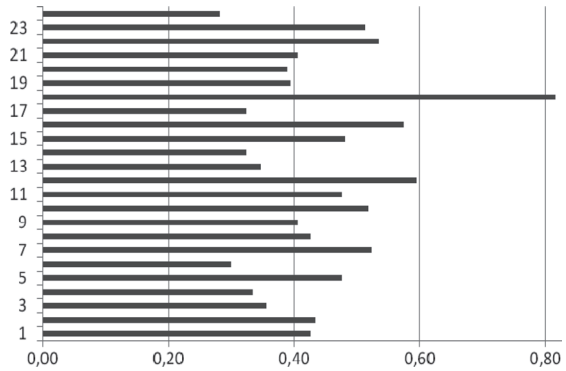


Рис. 2. Квалификация каждого из 24 программистов, усредненная по 16 технологиям

$$Qualif^{avg}(g) = \sum_{p \in P_g} Qualif(p) \Big/ n_g \qquad (4)$$

Помимо средней квалификации $Qualif^{avg}(g)$, важнейшим параметром, характеризующим группу g, является квалификация $Qualif^{best}(g)$ по лучшим представителям, которая определяется формулами (5) и (6).

$$Qualif^{best}(g) = \begin{cases} 0, \text{ если } \exists t, oblgt(t) \text{ и } mxlevel(g,t) < RL_t^g \\ \dfrac{\sum_{t \in T} rank(t) \cdot mxlevel(g,t)}{MaxQualif}, \text{ иначе} \end{cases} \qquad (5)$$

$$mxlevel(g,t) = \max_{p \in P_g} \left(level(p,t) \right) \qquad (6)$$

Формула (5) использует следующие величины: $oblgt(t)$ – предикат, принимающий значение истина, если технология является обязательной для группы (таблица); $mxlevel(g, t)$ – уровень лучшего представителя группы g по технологии t; RL_t^g – пороговое значение уровня лучшего представителя по технологии t.

Согласно формуле (5), квалификация по лучшим представителям $Qualif^{best}(g)$ получает нулевое значение, если существует хотя бы одна обязательная для группы технология t, для которой $oblgt(t)$ истинно, а уровень $mxlevel(g, t)$ лучшего представителя группы g меньше порогового значения RL_t^g. Это объясняется тем, что группа не способна выполнять проекты без высококвалифицированных специалистов по ключевым технологиям.

Взвешенная квалификация $Qualif^{w}(g)$ группы g оценивается в виде суммы квалификации $Qualif^{best}(g)$ по лучшим представителям с весом λ и средней квалификации $Qualif^{avg}(g)$ с весом $1 - \lambda$:

$$Qualif^{w}(g) = \lambda \cdot Qualif^{best}(g) + (1 - \lambda) Qualif^{avg}(g). \qquad (7)$$

Взвешенная квалификация при $0 \le \lambda \le 1$ может принимать любое значение в диапазоне [0, 1]. Чем больше значение λ, тем больший вес придается квалификации по лучшим представителям, и наоборот, чем меньше значение λ, те больший вес придается средней квалификации программистов группы. Средняя квалификация отражает состояние дел на текущий момент. Квалификации по лучшим представителям показывает возможности роста членов команды, ориентирующихся и подтягивающихся к экспертам по технологиям, которые входят в состав группы и являются образцовым примером.

Очевидно, что группы с низкой квалификацией не могут быть признаны работоспособными, а участие таких групп в работах над проектом является необоснованным или, по крайней мере, спорным. Исключение появления таких групп формализуется понятием пороговой взвешенной квалификации:

$$Qualif(g) = \begin{cases} Qualif^{w}(g), \text{ если } Qualif^{w}(g) \ge RQ^g \\ 0, \text{ в противном случае.} \end{cases} \qquad (8)$$

Пороговое значение квалификации RQ^g рекомендуется предпочтительно выбирать из диапазона от 0.5 до 1.0 в зависимости от требований к проекту и технологиям его разработки, а также от ожидаемого качества будущих результатов проектирования. В дальнейшем под квалификацией группы программистов будем понимать пороговую взвешенную квалификацию.

Группа g называется избыточной по квалификации, если в ней найдется программист $p \in g$ такой, что $Qualif(g \setminus \{p\}) \ge Qualif(g)$. Другими словами, квалификация группы g после вывода из нее программиста p оказывается не ниже квалификации до вывода программиста. Такое может произойти при выполнении двух условий:

1. Программист p не является единственным лучшим представителем группы g ни по одной технологии $t \in T$.

2. Квалификация $Qualif(p)$ программиста p ниже средней квалификации $Qualif^{avg}(g)$ группы g.

Если множество программистов разбито на множество G групп, то общая квалификация всех групп оценивается в виде суммы пороговой взвешенной квалификации по всем группам:

$$Qualification(G) = \sum_{g \in G} Qualif(g) \qquad (9)$$

Общая квалификация групп $Qualification(G)$ принимает значение в диапазоне от $k \cdot RQ^g$ до k, где k – число групп.

Задача оптимизации размера и состава программистских групп

Определим оптимизируемый параметр, критерий оптимизации и допустимое множество

задачи оптимизации. Оптимизируемым параметром является суммарная квалификация всех групп *Qualification*(*G*). Целевую функцию запишем в виде:

$$\max_{G\in\Omega} \; Qualification(G) \qquad (10)$$

где Ω – множество всех возможных разбиений *G* программистов из множества *P* на группы.

Сформулируем допустимое множество через систему ограничений, описываемых в виде требования к программистам и группам программистов по владению технологиями и инструментами программирования.

Требование 1. Оно относится к минимально-пороговому уровню RL_t^p владения технологией *t* каждым программистом *p* из множества *P*. Требование к минимальному уровню по всем технологиям представим вектором $RL^p = \{RL_1^p, \ldots, RL_m^p\}$. Реальный уровень L_t^p определяется выражением:

$$L_t^p = rank(t)\cdot level(p,t) \qquad (11)$$

Требование 2. Оно относится к минимально-пороговому уровню RL_t^g владения технологией *t* лучшим представителем группы программистов $g \in G$. Для всех технологий минимально-пороговый уровень представляется вектором $RL^g = \{RL_1^g, \ldots, RL_m^g\}$. Реальный уровень L_t^g владения лучшим представителем группы *g* технологией *t* определяется выражением:

$$L_t^g = rank(t)\cdot \max_{p\in P_g}\left(level(p,t)\right) \qquad (11)$$

Требование 3. Оно относится к минимально-пороговой взвешенной квалификации RQ^g каждой группы программистов $g \in G$. Требуемая минимальная квалификация одинакова для всех групп. Реальная взвешенная квалификация $Qualif^w(g)$ группы оценивается формулой (7).

Решение задачи оптимизации генетическим алгоритмом

Генетический алгоритм (GA) реализует случайный процесс эволюции популяции хромосом с целью нахождения наилучшего по совокупной квалификации разбиения программистов на группы. Хромосома строится из генов, соответствующих программистам. Значение гена есть номер группы, в которую включается программист. Фитнес функцией является общая квалификация *Qualification*(*G*). Генетическая операция скрещивания рекомбинирует участки двух хромосом и переводит программистов из одних групп в другие группы в хромосомах-потомках. Ее выполнение может привести к исчезновению групп в хромосомах-потомках. Нормализация хромосом выполняется перенумерацией групп. Операция мутации переводит случайным образом одного или больше программистов в другие группы. Операция селекции выбирает родителей по правилу рулетки и выбирает хромосомы для замены худших родителей в популяции. Хромосома с наибольшим значением функции полезности является решением оптимизационной задачи.

Результаты вычислительных экспериментов

Для проведения вычислительных экспериментов использовано разработанная программа, реализующая генетический алгоритм. Установлены следующие требования: ограничение снизу на квалификацию RL^p = {0.1, 0, 0.1, 0, 0.3, 0.25, 0, 0.15, 0.6, 0, 0, 0.25, 0, 0.2, 0.4, 0.2} программиста по каждой из 16 технологий, перечисленных в таблице; ограничение снизу на квалификацию RL^g = {0.2, 0, 0.2, 0, 0.4, 0.3, 0, 0.3, 0.75, 0.4, 0, 0.5, 0, 0.3, 0.5, 0.25} группы программистов по каждой из 16 технологий; ограничение снизу на пороговую взвешенную квалификацию $RQ^g = 0.5$ группы программистов по всем технологиям; вес $\lambda = 0.7$ квалификации группы по лучшим представителям во взвешенной квалификации группы.

Результаты экспериментов представлены на рис. 3–7. Рис. 3–5 показывают динамику работы генетического алгоритма в процессе эволюции популяции хромосом. Суммарная по всем группам пороговая взвешенная квалификации выросла (рис. 3) с 3.5 до 5.04 на протяжении 25 поколений. Рост составил около 30%. За это время число групп, удовлетворяющих всем выдвинутым требованиям по квалификации, увеличилось с 5 до 8 (рис. 4, сплошная линия), а число включенных в них программистов возросло с 15 до 23 (рис. 5, сплошная линия). Число групп, не удовлетворяющих требованиям, колебалось от 1 до 2 (рис. 4, пунктирная линия), а число попавших в них программистов колебалось от 9 до 1 (рис. 5, пунктирная линия). В итоге, эволюционный процесс сопровождался устойчивым ростом суммарной

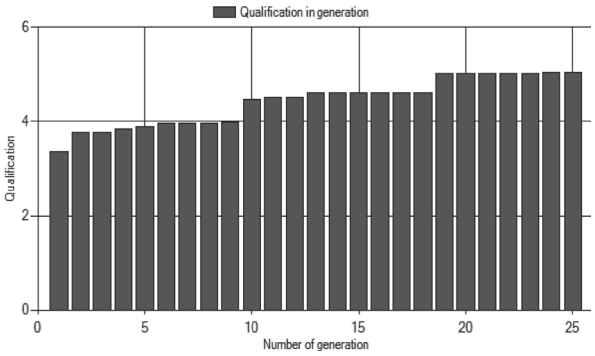


Рис. 3. Зависимость суммарной по всем группам пороговой взвешенной квалификации *Qualification*(*G*) от номера поколения в GA

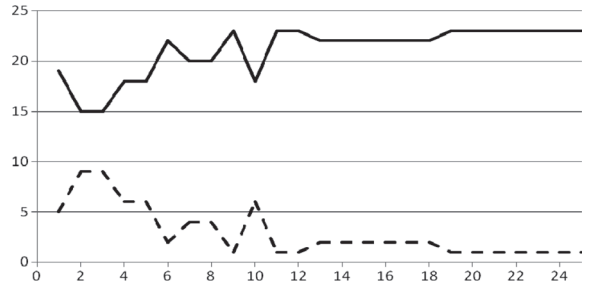


Рис. 5. Число программистов в группах, удовлетворяющих требованиям (сплошная), и в резерве (пунктирная) в зависимости от номера поколения в GA

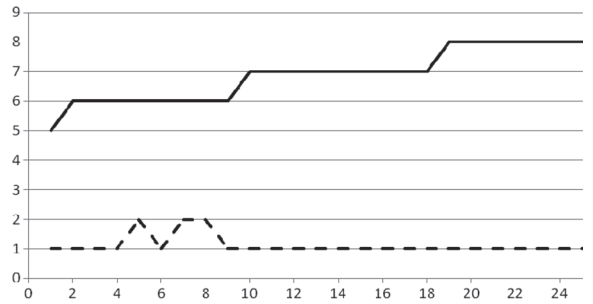


Рис. 4. Число групп, удовлетворяющих требованиям (сплошная) и неудовлетворяющих требованиям (пунктирная), в зависимости от номера поколения в GA

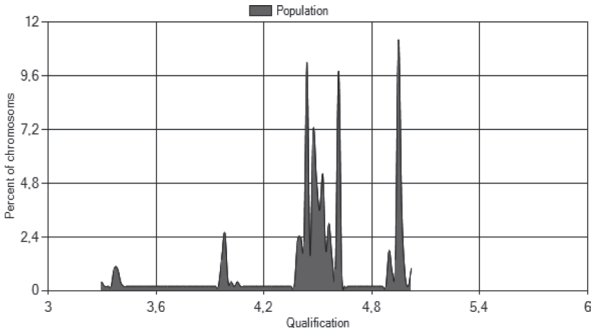


Рис. 6. Доля в % числа хромосом с лучшим значением функции полезности в популяции в зависимости от номера поколения в GA

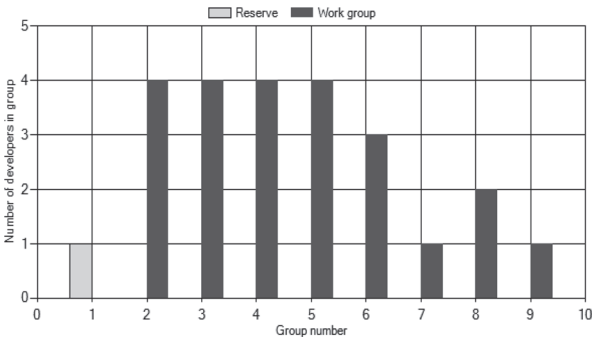


Рис. 7. Распределение программистов по группам в оптимальном решении (группа 1 – резерв)

пороговой взвешенной квалификации групп, представленных лучшими хромосомами, доля которых от размера популяции показана на рис. 6.

Лучшее распределение программистов по группам представлено на рис. 7. Всего сформировано 8 групп с суммарным уровнем квалификации 5.04 и средним уровнем 0.63, из которых 4 группы включают по 4 программиста, одна – 3 программиста, одна – 2 программиста, и две группы включают по 1 программисту. Неиспользованным (в резерве) оказался только один программист.

Заключение

Предложен метод оценки квалификации групп разработчиков по уровню знания и владения технологиями и инструментами программирования. Разработан генетический алгоритм оптимизации размера, состава и количества групп, максимизирующий суммарную квалификацию групп и минимизирующий число незадействованных в них программистов. Дальнейшие исследования будут направлены на интеграцию факторов квалификации и производительности групп программистов.

Литература

1. **Joshi, S.** Agile Development - Working with Agile in a Distributed Team Environment / S. Joshi // MSDN Magazine, 2012, Vol.27, No.1, pp.1–6.

2. **Прихожий, А. А.** Конспект лекций по дисциплине «Моделирование и оптимальное проектирование технических систем» / А.А. Прихожий // БНТУ, кафедра ПОВТиАС, 2013, с. 58–69.

3. **Barricelli, N. A.** Symbio genetic evolution processes realized by artificial methods / N.A. Barricelli // Methodos, 1957, pp. 143–182.

4. **Müller, J. P., Rao, A. S., Singh, M. P.** A-Teams: An Agent Architecture for Optimization and Decision-Support, Proceedings 5th International Workshop, ATAL'98 Paris, France, July 4–7, 1998, pp. 261-276.

5. **Прихожий, А. А.** Эволюционный метод оптимизации состава групп разработчиков с целью сокращения затрат и времени на выполнение проекта / А.А. Прихожий, А.М. Ждановский // Материалы научно-технической конференции «Информационные технологии в технических и социально-экономических системах», Минск, РИВШ, 2016. – С. 16-20.

6. **Red Monk** [Электронный ресурс] / сайт аналитической компании Red Monk. – Режим доступа: http://redmonk.com/sogrady/2016/07/20/language-rankings-6-16/. – Дата доступа: 26.02.2017.

7. **Cass, S.** The 2016 Top Programming Languages [Электронный ресурс] / IEEE Spectrum, 2016. – Режим доступа: http://spectrum.ieee.org/ computing/software/the-2016-top-programming-languages. – Дата доступа: 26.02.2017.

References

1. **Joshi, S.** Agile Development - Working with Agile in a Distributed Team Environment / S. Joshi // MSDN Magazine, 2012, Vol.27, No.1, pp.1–6.

2. **Prihozhy, A.** Lecture notes on "Modeling and Optimization for Engineering Systems Design" / A. Prihozhy // BNTU, Software for Computers and Automated Systems Dpt., 2013, pp. 58–69.

3. **Barricelli, N. A.** Symbio genetic evolution processes realized by artificial methods / N.A. Barricelli // Methodos, 1957, pp. 143–182.

4. **Müller, J. P., Rao, A. S., Singh, M. P.** A-Teams: An Agent Architecture for Optimization and Decision-Support, Proceedings 5th International Workshop, ATAL'98 Paris, France, July 4–7, 1998, pp. 261–276.

5. **Prihozhy, A.** Evolutionary Method of Software Teams Optimization for Reducing Time and Resources of Project Execution / A. Prihozhy, A. Zhdanouski // Proc. Conf. "Information Technologies in Engineering and Business", Minsk, RIHS, 2016, pp.16–20.

6. **Red Monk** [Электронный ресурс] / сайт аналитической компании Red Monk. – Режим доступа: http://redmonk.com/sogrady/2016/07/20/language-rankings-6-16/. – Дата доступа: 26.02.2017.

7. **Cass, S.** The 2016 Top Programming Languages [Электронный ресурс] / IEEE Spectrum, 2016. – Режим доступа: http://spectrum.ieee.org/computing/software/the-2016-top-programming-languages. – Дата доступа: 26.02.2017.

Поступила	*После доработки*	*Принята к печати*
20.03.2018	*12.04.2018*	*01.06.2018*

Prihozhy A., Zhdanouski A.

METHOD OF QUALIFICATION ESTIMATION AND OPTIMIZATION OF PROFESSIONAL TEAMS OF PROGRAMMERS

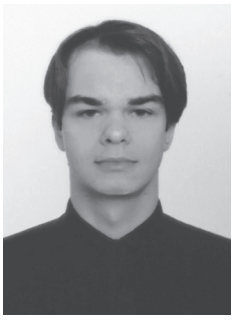
prihozhy@yahoo.com

The problem of building and optimizing the teams of programmers taking into account the qualification and the level of skills in programming technologies and tools is considered. Known technological environments for the management of teamwork, such as Agile, formulate only general requirements and principles for building the teams and the distribution of work between them. A method for formalizing and evaluating the qualification of individual programmers and entire groups of pro-grammers has been proposed. The input data are attributes of the most popular technologies and programming tools, includ-ing technology rating, as well as the results of a survey of programmers on their level of skills. The qualification of a group of programmers is evaluated taking into account the requirements for a particular project, which integrates three components: the average qualification of programmers included in the group; the qualification of the group with respect to the best repre-sentatives for each of the technologies; threshold values of the levels of programmer qualification and group qualification for each of the technologies, as well as threshold values of the integrated qualification, reflecting the specifics of the given project. The contribution of each component is determined by appropriate weights. The proposed method is a basis for a genetic algo-rithm that performs the search for the number, size and staff of groups of programmers which yield a maximum of total quali-fication. Software has been developed and computer experiments have been carried out on a set of programmers who graduat-ed from Belarusian universities. The obtained results demonstrate the real ability of the system to find the number, size and staff of groups of programmers, which increase the overall qualification of groups by 30% while minimizing the number of unemployed developers. The results prove the practical importance of the method and software in the field of technologies and tools for the management of professional teams of programmers.

Keywords: *Programmer, technology, tool, qualification, skills, programmer team, team staff, optimization.*

Прихожий Анатолий Алексеевич – профессор кафедры программного обеспечения вычислительной техники и автоматизированных систем БНТУ, доктор технических наук (1999), профессор (2001). Научные интересы в области языков программирования и описания цифровой аппратауры, распараллелтвающих компиляторов, инструментальных средств проектирования программных и аппаратных систем на логическом, поведенческом и системном уровнях. Имеет более 300 научных публикаций в Восточной и Западной Европе, США и Канаде.

Anatoly Prihozhy is a full professor at the Computer and system software department of Belarusian national technical university, doctor of science (1999) and professor (2001). His research interests include programming and hardware description languages, parallelizing compilers, and computer aided design tools for software and hardware at logic, high and system levels. He has over 300 publications in Eastern and Western Europe, USA and Canada.



Ждановский Арсений Матвеевич – аспирант кафедры «Программное обеспечение вычислительной техники и автоматизированных систем» БНТУ, инженер программист компании «EPAM Systems», научные интересы в области языков и технологий программирования и методов оптимизации.

Zhdanouski Arseni is a postgraduate of the Computer and system software department of Belarusian national technical university, and a software engineer at EPAM Systems. His research interests include programming languages and techlogies and methods of optimization**.**