УДК 004.4

DOI: 10.18372/2073-4751.77.18664

Русанова О.В., к.т.н.,
orcid.org/0000-0003-0145-3012,
e-mail: olga.rusanova.v@gmail.com
Корочкін О.В., к.т.н.,
orcid.org/0000-0002-6569-5849,
e-mail: avcora@gmail.com,
Ачілов А.В.,
e-mail: andrii.achilov@gmail.com

СПОСІБ УПРАВЛІННЯ ПРОЄКТАМИ НА БАЗІ ОЦІНОК STORY POINTS

**Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»**

Вступ

Управління проєктами сьогодні є стратегічним засобом для ефективного контролю за ефективністю роботи над проєктом та його витратами. В епоху третьої цифрової революції, дедалі більша кількість систем піддається автоматизації. У цьому контексті системи управління проєктами є частиною загального тренду: вони дозволяють зменшити час, витрачений на організацію і управління проєкту, завдяки корисним функціям, які вони надають. Це, зокрема, можливість відслідковувати прогрес співробітників, терміни виконання задач і всього проєкту, а також визначення критичного шляху проєкту. До того ж, присутність Інтернету тільки покращує взаємодію учасників процесу, цілком витісняючи керування проєктами "вручну".

Сучасні тенденції диктують нові умови в процесі розробки програмного забезпечення. Поява гнучких методологій, зокрема *Scrum*, спричинила вагомі зміни в IT-індустрії і вивела уперед ті компанії, які прийняли і застосували нову концепцію у своїй діяльності. *Scrum* приніс не лише продуктивність і ефективність в розробці програмного забезпечення, а і нові стандарти в проєктуванні.

Завдяки *Scrum*, здобула популярність оцінка задач проєкту не в людино-годинах, а у *Story Points*. Вона означає не обсяг часу для виконання задачі, а кількість зусиль, яку необхідно потратити на неї розробнику [1]. Причому, ця оцінка незалежна від досвіду розробника, що дозволяє менеджерам проєктів гнучко оперувати задачами і визначенням строків проєкту.

Ринок програмних засобів пропонує широкий вибір серед додатків, що дозволяють керівникам проєктів і/або керівникам компаній та підприємств відслідковувати роботу над проєктами. Здебільшого, взаємодія з ними зводиться до наступного: відповідальна людина вводить інформацію про задачі, їх взаємозв'язки та співробітника, який має її виконати. Також за потреби призначаються додаткові обмеження по часу виконання чи бюджету. Після отримання інформації про проєкт, додатки надають аналітичну інформацію про статус проєкту, затримки виконання задач та термін виконання проєкту. В табл. 1 наведено порівняльний аналіз функціоналу деяких популярних відомих систем управління проєктами.

Як видно, більшість мають обширний функціонал і навіть використовують метод критичного шляху (*CPM*), але їм бракує реалізації автоматичного планування проєктів з використанням методу оцінки і перегляду планів (*PERT*), так як саме планування в них відбувається вручну, окрім додатку СУПОМП, який описаний у роботі [2]. Крім того, жоден із додатків не використовує оцінку складності в *Story Points*. Частина додатків також надає можливість відслідковувати витрати на співробітників, але він розподіляється лише на задачі. У відомих додатках немає розділення співробітників за їх кваліфікацією (*velocity*), що мало б впливати на час

виконання задачі та оплату за неї, не розглядається також спеціалізація співробітників.

Постановка задачі

Завдання полягає в розробці системи управління проєктами, що використовуватиме нові алгоритми автоматизації розподілу задач проєктів між співробітниками з урахуванням зазначених вище недоліків відомих підходів.

Таблиця 1. Порівняльний аналіз функціоналу деяких популярних відомих систем управління проєктами

Функціональність	*Wrike*	*Microsoft Project*	*Monday work management*	*ClickUp*	*СУПОМП*
Гнучкість створення задач та опису характеристик	Так	Так	Так	Так	Так
Широкий вибір виглядів задач	Так	Обмежена кількість	Так	Так	Так
Підтримка коментарів	Так	Ні	Так	Так	Ні
Контроль статусу задач	Так	Ні	Так	Так	Так
Вигляд діаграми Ганта	Так (тільки в платній версії)	Так (тільки в платній версії)	Так (тільки в платній версії)	Так	Так
Мережевий графік	Ні	Так	Ні	Ні	Так
Автоматична розстановка задач на діаграмі Ганта	Так	Так	Ні	Ні	Так
Перегляд критичного шляху проєкту	Так	Так	Так	Так	Так
Налаштування доступу до об'єктів	Так	Ні	Так	Так	Так
Перегляд аналітики за проєктами	Так	Так	Так	Так	Так
Реалізація методу PERT	Ні	Так	Ні	Ні	Так
Відслідковування зайнятості співробітників	Так	Так, але не зручно	Так	Так	Так
Автоматичний планування робіт між співробітниками	Ні	Ні	Ні	Ні	Так
Підтримка оцінок задач в story points	Ні	Ні	Ні	Ні	Ні

Особливостями алгоритмів є планування задач, що мають оцінку складності в *Story Points* та врахування додаткових обмежень, які існують в реальних проєктах: кваліфікації співробітників та можливості виконання лише обмеженої підмножини задач певним співробітником. Реалізацією системи є веб-додаток, який на основі даних користувача надаватиме пропонований результат розподілу співробітників по задачах проєкту у вигляді діаграми Ганта та додаткових числових результатах – часі виконання та кількості залучених співробітників.

При плануванні пропонується використовувати один з наступних критеріїв оптимізації: 1) виконання проєкту за мінімальний час при заданому складі команди; 2) мінімальну кількість людей у команді, яка виконає проєкт за обмежений (заданий) час.

Мета

Метою даної роботи є мінімізація часу та вартості розробки проєктів із задачами на базі оцінки *Story Points* за рахунок запропонованих способів оптимізації на базі мережевої моделі.

Основна частина

Сучасні інструменти планування проєктів використовують методи мережевого планування, що базуються на методах CPM та PERT. Ці методи дозволяють відповісти на наступні питання: 1) скільки часу необхідно для виконання всього

проєкту? 2) коли повинні закінчуватися і починатися окремі роботи? 3) які роботи є критичними і повинні виконуватися в першу чергу за графіком, щоб не допустити зрив термінів виконання проєкту? 4) на який термін можна відкласти виконання некритичних робіт, щоб це не вплинуло на терміни виконання проєкту?

Методи мережевого планування використовують певні поняття і характеристики, що впливають на процес планування і визначені у роботах (критичнй шлях, ранній та пізній початки виконання задач, раннє та пізнє завершення виконання задач, резерв часу)[3].

У мережевому плануванні завдання та залежності між ними зображують у вигляді мережевого графу. У роботі [2] для зображення мережевого графу використовується метод діаграми пріоритетів (PMD, precedence diagram method), де вершинам відповідають задачі проєкту, а дугам зв'язки між задачами [3]. Кожна вершина має свій номер і вагу (час виконання) задачі. Приклад такого графу зображено на рис.1 (всередині вершин вказано час виконання задачі, а справа зверху – індекс вершини).

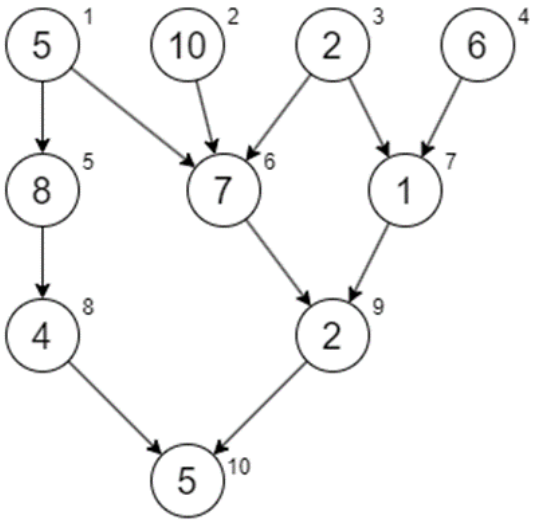

Рис. 1. Приклад графу задач проєкту для додатку СУПОМП

Для автоматичного розподілу задач між співробітниками за мінімальний час у роботі [2] розраховувались пріоритети усіх задач на основі критичних шляхів від кожної з них до кінця графу задачі, а далі задачі розподілялись по співробітниках відповідно до їх пріоритетів та з урахуванням зв'язків між задачами.

На основі отриманих результатів була побудована діаграму Ганта, яка наочно демонструвала графік виконання робіт проєкту.

У даній роботі пропонується модифікувати вищезазначений підхід до мережевого планування на основі зміни вхідних даних і відповідних алгоритмів.

1. У графі задачі в якості ваги вершин (задач) використовується оцінка складності в *Story Points*.

2. Для кожного співробітника задається його власна швидкість розробки (velocity), що вимірюється в *Story Points* за тиждень. Це значення визначається в Scrum-командах, як суму складностей виконаних задач співробітником за попередній тиждень, або як середнє значення за тиждень за інший період, наприклад, попередній спринт. Маючи інформацію про складність задачі та швидкість розробки певного співробітника можна розрахувати тривалість виконання задачі за наступною формулою:

$$t = \frac{P}{v} * d, \qquad (1)$$

де t – тривалість виконання задачі, P – складність задачі в *Story Points*, v – швидкість розробки співробітника, d – кількість робочих днів в тижні.

3. Задається зарплата кожного співробітника за тиждень.

4. Задається спеціалізація кожного співробітника, а саме, які задачі в проєкті він може виконувати.

5. В алгоритмах планування розглядаються лише ранні початок та завершення задач. При плануванні проєкту часто замовник і не знає наскільки довго може тривати проєкт, але він точно хоче, щоб він тривав тривати якнайменше. Оскільки часто немає точного орієнтиру завершення проєкту, то розрахунок пізніх початку та завершення задач неможливі та і непотрібні в даному випадку. Також, це спростить обрахунок моментів часу завершення задач кожним із співробітників.

6. Змінюється поняття пріоритету задач. Пріоритетом задачі вважатимемо величину критичного шляху до вершини в мережевому графіку. Критичний шлях до вершини – це найбільша сума складностей задач від поточної вершини до кінцевої. В такому разі, критичний шлях для початкової задачі дорівнюватиме критичному шляху мережі, а для кінцевої – значенню складності цієї задачі.

Пріоритет задачі розраховується за наступною рекурсивною функцією від складності задачі (2) :

$$f(W_i) = \begin{cases} W_i + max\left(f\left(W_{i_1}\right), f\left(W_{i_2}\right), \dots, f\left(W_{i_n}\right)\right), \text{якщо } n > 0 \\ W_i, \text{якщо } n = 0 \end{cases}, \quad (2)$$

де W_i – складність (вага) i -ої задачі, $W_{i_1}, W_{i_2}, \dots, W_{i_n}$ – складності (ваги) дочірніх задач i -ої задачі, n – кількість нащадків i-тої задачі ($n \geq 0$).

На основі формули (2) можна визначити пріоритет задачі, враховуючи значення пріоритетів дочірніх задач наступним чином:

$$P_i = \begin{cases} W_i + \max\left(P_{i_1}, P_{i_2}, \dots, P_{i_n}\right), \text{якщо } n > 0 \\ W_i, \text{якщо } n = 0 \end{cases}, \quad (3)$$

де P_i – пріоритет i -тої задачі, W_i – складність (вага) i -ої задачі, $P_{i_1}, P_{i_2}, \dots, P_{i_n}$ – пріоритети дочірніх задач i-ої задачі, n – кількість нащадків i-тої задачі ($n \geq 0$).

Пріоритет задач впливатиме на те, яку задачу краще виконувати наступною із наявних. Через те, що для кожної вершини пріоритет враховує не тільки власну складність, а й складність всіх наступних задач до кінцевої, більший пріоритет вершини вказуватиме на те, що вона знаходиться на більш довгому шляху. Такий вибір вершини допоможе ефективніше розподіляти ресурси у вигляді співробітників та коштів на проєкті. Для графу на рис.1 (де числа всередині вершин є складність в *Story Points*) критичний шлях виглядатиме наступним чином (рис. 2).

Числа зліва зверху вершин вказують на критичний шлях від поточної вершини до останньої в графі. Для даного графу, критичний шлях проєкту становить 24.

Алгоритм визначення плану виконання проєкту за мінімальний час при заданому складі співробітників.

Для даного алгоритму потрібно мати наступні вхідні дані: граф задач проєкту, список доступних співробітників проєкту для кожної із задач, список швидкостей розробки співробітників та список їх зарплати за тиждень. У графі задач вершинами є власне задачі, а ребрами – їх зв'язки між собою; зв'язок між задачами 1 і 2 означає, що задача 2 може виконуватися тільки після того, як завершилась задача 1. Також, кожна вершина має свою вагу – складність виконання задачі. Приклад такого графу зображено на рис. 1.

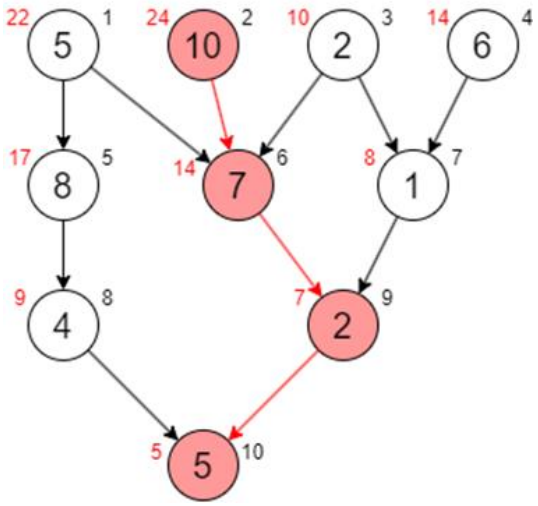

Рис. 2. Критичний шлях графа

Традиційно, структура звязків графу задачі може бути задана у вигляді матриці суміжності. Крім того задається список ваг вершин. Для графа на рис.1 матриця суміжності і список ваг вершин матимуть наступний вигляд (рис. 3, табл. 2).

	1	2	3	4	5	6	7	8	9	10
1	0	**1**	**1**	0	0	0	0	0	0	0
2	0	0	0	**1**	0	0	0	0	0	0
3	0	0	0	0	0	**1**	**1**	0	0	0
4	0	0	0	0	**1**	**1**	0	0	0	0
5	0	0	0	0	0	0	0	**1**	0	0
6	0	0	0	0	0	0	0	0	**1**	0
7	0	0	0	0	0	0	0	0	**1**	0
8	0	0	0	0	0	0	0	0	0	**1**
9	0	0	0	0	0	0	0	0	0	**1**
10	0	0	0	0	0	0	0	0	0	0

Рис. 3 Матриця суміжності
для графу задач (рис. 1)

Таблиця 2. Список ваг вершин

Номер вершини	*Складність*
1	5
2	4
3	2
4	2
5	10
6	4
7	6
8	3
9	2
10	7

Приклад списку доступних співробітників показано в табл. 3.

Таблиця 3. Приклад списку доступних співробітників

Ім'я співробітника	*Зарплата (за тиждень)*	*Швидкість розробки (в story points за тиждень)*
Співробітник 1	150$	15
Співробітник 2	100$	10
Співробітник 3	30$	3
Співробітник 4	20$	2

Після отримання вхідних даних, спочатку сформуємо список задач за спаданням їх пріоритетів, визначених за формулою (3). Виникає питання – чи не порушить такий порядок пріоритетів задач вимоги, що дочірня задача не повинна виконуватись раніше за батьківську задачу? Покажемо, що це не так.

Відповідно до формули (3) при $n > 0$ маємо:

$$P_i = W_i + \max\left(P_{i_1}, P_{i_2}, \dots, P_{i_n}\right). \quad (4)$$

Нехай $P_{max} = \max\left(P_{i_1}, P_{i_2}, \dots, P_{i_n}\right)$, $n > 0$. Звідси отримуємо n нерівностей:

$$P_{max} \geq P_{i_1}, P_{max} \geq P_{i_2}, \dots, P_{max} \geq P_{i_n}, \quad n > 0. \quad (5)$$

Також з формули (4) отримаємо, що $P_i = W_i + P_{max}$. Оскільки $W_i \geq 0$ (оскільки оцінка в *Story Points* є невід'ємним числом), то $P_i \geq P_{max}$.

На основі співвідношень (4) та (5) маємо наступні нерівності :

$$P_i \geq P_{i_1}, P_i \geq P_{i_2}, \dots, P_i \geq P_{i_n}, n > 0. \quad (6)$$

Таким чином, відповідно до нерівностей (6) пріоритет батьківської задачі P_i не менший за пріоритет дочірніх задач, а отже батьківська задача буде йти в списку раніше за дочірню. При однакових пріоритетах ми зважатимемо на номер задачі (батьківська задача матиме менший номер ніж дочірня, тому вона, знову таки, йтиме у списку раніше за дочірню).

Перейдемо безпосередньо до самого алгоритму. Сформуємо список зайнятості співробітників, де кожен елемент спочатку дорівнюватиме нулю. Алгоритм працює ітераційно. На кожному кроці визначається кількість задач, які готові до виконання. У порядку спадання їх пріоритетів для кожної i-ої задачі зі списку розраховуємо час її початку та завершення при її виконанні кожним із доступних співробітників для цієї задачі. Час початку задачі визначається за формулою (7) і розраховується як макимум із зайнятості співробітника та часом завершення найпізнішої батьківської задачі. Для задач, що не мають попередників час початку задачі дорівнюватиме зайнятості співробітника. Час завершення задачі визначається за формулою (8) і дорівнює сумі часу початку задачі і її тривалості.

$$t_{\text{п}(i,j)} = \begin{cases} max\left(B_j, t'_{\text{з}i_1}, t'_{\text{з}i_2}, \dots, t'_{\text{з}i_n}\right), n > 0 \\ B_i, n = 0 \end{cases}; \quad (7)$$

$$t_{\text{з}(i,j)} = t_{\text{п}(i,j)} + \frac{P_i}{v_j} * d, \quad (8)$$

де $t_{\text{п}(i,j)}$ – час початку i-ої задачі при її виконанні j-им працівником, B_j – зайнятість j-ого працівника, $t'_{\text{з}i_1}, t'_{\text{з}i_2}, \dots, t'_{\text{з}i_n}$ – моменти часу завершення батьків i-ої задачі, n – кількість батьків i-ої задачі, $t_{\text{з}(i,j)}$ – час завершення i-ої задачі при її виконанні j-им працівником, P_i – пріоритет i-ої задачі, v_j – швидкість розробки j-ого працівника, d – кількість робочих днів в тижні. Ми розглядатимемо стандартний 5-денний робочий тиждень, тому у формулі (8) $d = 5$.

Далі, на тому ж кроці визначаємо найкращого виконавця j для i-ої задачі за меншим часом завершення. При однаковому часі завершення перевагу віддаємо співробітнику з мінімальною зарплатою для здешевлення проєкту. Початком і завершенням виконання задачі i стане, відповідно, час початку і завершення виконання задачі i кращим виконавцем. Оновимо також зайнятість цього виконавця – вона дорівнюватиме часу завершення виконання задачі i. Після цього, можна переходити до наступного кроку ітерації.

Завершенням алгоритму є визначення співробітника для останньої із списку задачі проєкту. Час її завершення і буде часом виконання всього проєкту в цілому. Знайдені за час роботи алгоритму моменти часу початку і завершення задач та призначених виконавців на кожну із задач будуть результатами планування виконання проєкту, які можна буде зобразити на діаграмі Ганта.

Для проєкту, граф якого зображено на рис. 1, і списку співробітників в табл. 3 діаграма Ганта за результатами планування виконання задач виглядатиме наступним чином (рис. 4).

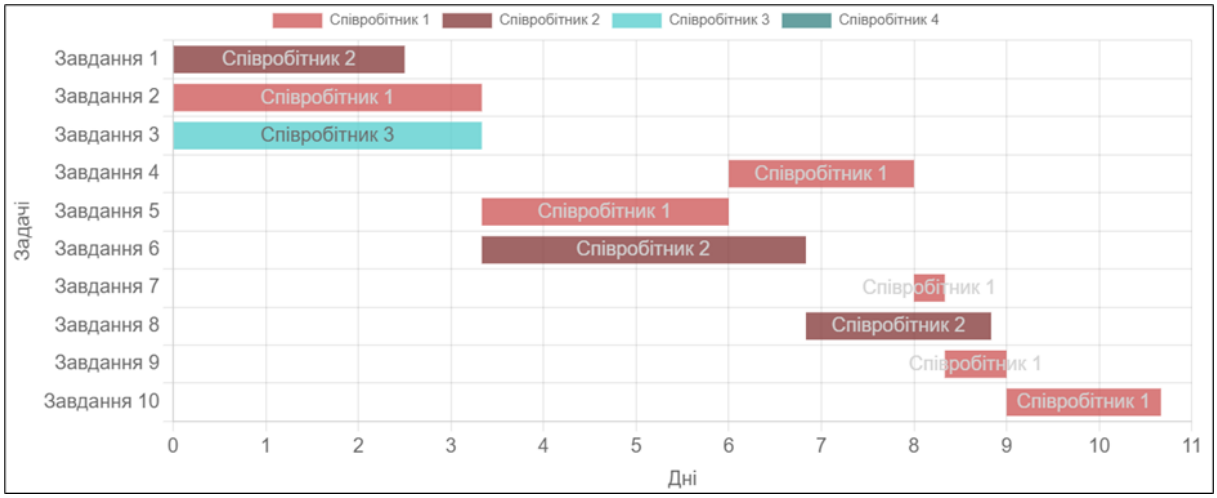

Рис. 4. Діаграма Ганта за результатами планування виконання задач проєкту

Алгоритм визначення мінімальної кількості співробітників, яка виконає проєкт за обмежений (заданий) час.

У попередньому алгоритмі для визначення плану проєкту ми використовували усіх співробітників. Для випадку, коли час на проєкт доволі малий, для його економії доводиться використовувати усіх співробітників, тому що вони можуть виконувати деякі задачі паралельно. Однак із зменшенням кількості людей, час, витрачений на проєкт зростає. При обмеженій кількості людей і маючи вибір кого обрати на проєкт, менеджеру логічно було б обрати тих людей, які зроблять роботу

якнайшвидше, тобто обирати найбільш кваліфікованих розробників.

У даній роботі пропонується алгоритм для знаходження мінімальної кількості людей в команді для виконання проєкту за заданий час, використовуючи ітераційно попередній розглянутий алгоритм. Ми будемо поступово обмежувати кількість співробітників, які виконують задачі, вилучаючи найповільнішого, потім повільнішого з тих, що лишилися і так далі, поки виконання проєкту не досягне очікуваного часу. Тоді остання успішна конфігурація співробітників і буде отриманим результатом. В алгоритмі також враховується той факт, що не всі люди в команді можуть виконувати певні задачі, а тому не будь-який співробітник може бути вилучений з команди.

Розглянемо приклад роботи даного алгоритму для проєкту, граф якого зображено на рис. 1, для списку співробітників в таблиці 3, а також при обмеженні часу виконання проєкту за 11 днів. Результати даного алгоритму представлені у табл. 4. Перша ітерація співпадає з ітерацією попереднього алгоритму, бо використовує ті ж дані. Наступні ж ітерації використовують меншу початкову кількість співробітників. Час виконання проєкту на третій ітерації такий же як і на першій та другій через те, що ті задачі, які виконував Співробітник 3 мали достатній запас часу для їх виконання менш кваліфікованим співробітником 2. Таким чином, мінімальна кількість співробітників, які забезпечують час виконання проєкту за 11 днів дорівнює двом.

Таблиця 4. Результати алгоритму

Ітерація	*Доступні співробітники*	*Час виконання проєкту*	*Використані співробітники*
1	Співробітник 1, Співробітник 2, Співробітник 3, Співробітник 4	10,67 днів	Співробітник 1, Співробітник 2, Співробітник 3
2	Співробітник 1, Співробітник 2, Співробітник 3	10,67 днів	Співробітник 1, Співробітник 2, Співробітник 3
3	Співробітник 1, Співробітник 2	10,67 днів	Співробітник 1, Співробітник 2
4	Співробітник 1	16,67 днів	Співробітник 1

Висновки

У роботі було виконано огляд теоретичних відомостей про сучасний стан управління проєктами, нові підходи до управління, та було виконано короткий огляд існуючих систем управління проєктами. На підставі аналізу було виявлено відсутність автоматизації планування проєктів і невикористання сучасного підходу до врахування оцінок складностей задач проєкту в *Story Points*.

Було запропоновано новий підхід до автоматизації планування на основі існуючих мережевих методів, але з модифікаціями, які дозволяють врахувати кваліфікацію співробітників, той факт, що не всі задачі доступні для виконання співробітниками, а також враховано оцінювання складності задач в *Story Points*. В процесі розробки було використано два критерії оптимізації: мінімальний час виконання проєкту та мінімальна кількість співробітників для планування за заданий час.

Запропоновані способи розподілу задач проєкту між співробітниками збільшують точність планування, оскільки є більш наближеним до реалістичних сценаріїв, ніж традиційні підходи.

На основі запропонованих підходів розроблений додаток, що реалізує представлені алгоритми, який може бути застосований для підприємств, де команди розробки працюють із використанням Scrum-підходу, зокрема, але не обмежуючись, в IT-компаніях.

Література

1. Devisdon D. Why do we use Story Points for Estimating? URL:

https://www.scrum.org/resources/blog/why-do-we-use-story-points-estimating.

2. Русанова О. В., Корочкін О. В., Медведкова Ю. І. Спосіб управління проєктами на основі мережевого планування. *Проблеми інформатизації та управління*. 2022. №3(71). С. 51–56.

3. Єгорченков О. В., Єгорченкова Н. Ю., Катаєва Є. Ю. Азбука управління проектами. Планування : навч. посіб. Київ : КНУ ім.Т.Шевченка, 2017. 117 с.

Русанова О.В., Корочкін О.В., Ачілов А.В.

СПОСІБ УПРАВЛІННЯ ПРОЄКТАМИ НА БАЗІ ОЦІНОК STORY POINTS

Робота присвячена розробці методів побудови систем управління проєктами, що дозволяє виконувати автоматичне планування проєктів, задачі в яких оцінюються в story points. У роботі було виконано огляд існуючих систем і запропоновано підхід до автоматизації планування проєктів за мінімальний час їх виконання, а також з мінімальною кількостю співробітників, яка виконує проєкт за заданий час.

Запропонований підхід дозволяє враховувати кваліфікацію, спеціалізацію та зарплату співробітників. Врахування цих особливостей, поряд із використанням оцінки складності задач в story points, наближає планування до реальних умов, що дозволяє підвищити точність планування, скоротити час виконання проєкту і/або витрати на співробітників і проєкт в цілому.

***Ключові слова**: проєктне управління; мережеве планування; планування виконання задач; Story Points; оптимізація часу виконання; оптимізація кількості співробітників.*

Rusanova O.V., Korochkin O.V., Achilov A.V.

METHOD OF PROJECT MANAGEMENT BASED ON STORY POINTS ESTIMATIONS

The paper is devoted to the development of building project management systems methods that allows to perform automatic project scheduling, where tasks are estimated in story points. The paper reviewed the existing systems and proposed an approach to the automation of project scheduling in minimum time, and using the minimum employees number who will complete the project in the given time.

The proposed approach allows taking into account the velocity, specialization and salaries of employees. Taking into account these features, and using of the tasks complexity in story points, brings scheduling closer to reality, which allows to increase the accuracy of scheduling, reduce the time of project execution and/or costs for employees and the project in general.

***Keywords**: project management; network scheduling; tasks execution scheduling; Story Points; optimization of execution time; optimization of the employees number.*