ԾՐԱԳՐԱՅԻՆ ԱՊԱՀՈՎՄԱՆ ՆԱԽԱԳԾԵՐԻ ԿԱՌԱՎԱՐՄԱՆ ՄԵԹՈԴՆԵՐԻ ՀԱՄԵՄԱՏԱԿԱՆ ՎԵՐԼՈՒԾՈՒԹՅՈՒՆ

Կ. Վ. ՊԵՏՐՈՍՅԱՆ, Ա. Հ. ԳՐԻԳՈՐՅԱՆ

Ծրագրային ապահովման մշակման ներկայիս վիճակը բավականին հեռու է կատարյալ լինելուց: Մշակվող համակարգերի զգալի մասը ավարտին են հասցվում լավագույն դեպքում վերջնաժամկետից ավելի ուշ, բյուջեի գերածախսով, իսկ արտադրանքն էլ չի բավարարում պատվիրատուի բոլոր սպասելիքները, որի պատճառով անընդհատ ենթարկվում է փոփոխությունների: Ծրագրային ապահովման փոփոխությունների պատճառ կարող են հանդիսանալ բիզնես պայմանների և միջավայրի փովոխությունները, մարդկանց սպասելիքները, տեխնոլոգիաների փոփոխությունները, տարբեր իրավիճակներից բխող անորոշությունները և այլն: Ծրագրային ապահովման ճարտարագիտության մեջ առկա են ծրագրային ապահովման մշակման մի շարք մեթոդաբանություններ, որոնք կարելի է բաժանել երկու խմբի՝ ավանդական և ճկուն: Ավանդական մեթոդաբանություններից են, օրինակ, ջրվեժային և պարուրաձև մոդելները, իսկ բյուրեղային, ծայրահեղ ծրագրավորման և դինամիկ համակարգերի մշակման մոդելները պատկանում են ճկուն մեթոդաբանությունների թվին: Համակարգի մշակման ավանդական մեթոդաբանությունների դեպքում (ինչպես օրինակ՝ ջրվեժային մեթոդաբանությունը), համակարգի պահանջները սահմանվում և հաճախ «սառեցվում» են նախագծի մշակման սկզբում: Դա նշանակում է, որ պահանջների փոփոխության արժեքը նախագծի ավելի ուշ նախատեսված փուլում, որը լայնորեն տարածված է իրական գործընթացներում, կարող է շատ բարձր լինել: Այս տեսակետից շատ կարևոր է ճկուն տեխնոլոգիաների կիրառումը ծրագրային ապահովման ճարտարագիտության մեջ: Ճկուն մեթոդների մեծամասնությունը փորձում է նվազագույնի հասցնել ռիսկը՝ մշակելով ծրագրային ապահովումը կարճ ժամանակահատվածներով, որոնք կոչվում են կրկնվող քայլեր:

Ավանդական մեթոդաբանության ջրվեժային մոդելն առաջարկված մոդելներից հնագույնն է: Հայտնի է այս մոդելի երկու տարատեսակ՝

- դասական ջրվեժային մոդել,
- ջրվեժային մոդել՝ հետադարձ կապով:

Մոդելը կոչվում է ջրվեժային, որովհետև նախագծի մշակումը դիտվում է որպես փուլերի հերթականություն, ընդ որում՝ անցումը հաջորդ՝ ավելի ցածր փուլի, տեղի է ունենում միայն ընթացիկ փուլի աշխատանքները լրիվ ավարտելուց հետո: Ենթադրվում է, որ մշակումը սկսվում է համակարգային մակարդակում և անցնում է վերլուծության, նախագծման, կոդավորման, թեստավորման և ներդրման փուլերը: Փաստորեն՝ մոդելավորվում են ստանդարտ ճարտարագիտական ցիկլի գործողությունները:

Այս մոդելի թերությունները որոշ չափով շտկված են՝ ձևափոխված, հետադարձ կապով ջրվեժային մոդելում, որն զգալիորեն ավելի ճկուն է և թույլ է տալիս փուլերի միջև որոշակի մոտարկումներ կատարել:

Արդի ծրագրային ապահովման ճարտարագիտության մեջ ջրվեժային մոդելը մեծ խնդիրների համար սահմանափակ կիրառություն ունի:

Պարուրաձև մոդելը հիմնվում է դասական ջրվեժային կենսական ցիկլի և նախատիպավորման ամենալավ հատկությունների վրա, որոնց գումարվում է ռիսկի վերլուծության նոր տարրը, որը բացակայում էր նախորդ մոտեցումներում:

Պարուրաձև մոդելը ծրագրային ապահովման մշակման կենսափուլի արդիականացված մոդել է, որը կենտրոնանում է նախագծի ռիսկերը նախապես որոշելու և նվազեցնելու վրա: Պարույր նախագիծը սկսում է փոքր մասշտաբով, բացահայտում է ռիսկերը, կազմում է ռիսկերի կառավարման պլան և որոշում է՝ արդյոք արժե ձեռնարկել նախագծի հաջորդ քայլը, այսինքն՝ իրականացնել պարույրի հաջորդ կրկնվող քայլը: Դրա արագ մշակման առավելությունը չի բխում նախագծի իրականացման արագության ավելացումից, այլ հետևում է նախա-

գծի ռիսկի մակարդակը շարունակաբար նվազեցնելուց, որն ազդում է նախագծի ավարտի համար պահանջվող ժամանակի վրա: Պարուրաձև մոդելի կիրառման հաջողությունը կախված է ճիշտ կառավարումից: Այն կարող է օգտագործվել նախագծերի մեծամասնության համար, որտեղ ռիսկի դիտարկումը միշտ շահավետ է: Պարուրաձև մեթոդաբանությունն ընդարձակում է ջրվեժային մոդելը նախատիպերի մշակման ներկայացմամբ: Ջրվեժային մոդելը հիմնականում փոխարինվում է պարուրաձև մոդելով բարդ նախագծեր մշակելու դեպքում [1]:

Տեղեկատվական տեխնոլոգիաների զարգացմանը զուգընթաց ճկուն մոդելների կիրառումը, ինչպիսին է ծայրահեղ ծրագրավորումը (Extreme Programming), ավելի նշանակալի է դառնում: Ծայրահեղ ծրագրավորումը շատ անկայուն միջավայրում ծրագրային ապահովում մշակելու մեթոդաբանություն է, որը ճկունության հնարավորություն է տալիս մոդելավորման գործընթացում: Ծայրահեղ ծրագրավորման հիմնական նպատակն է՝ նվազեցնել փոփոխության արժեքը ծրագրային ապահովման պահանջների փոփոխության ընթացքում: Ծայրահեղ ծրագրավորման հիմնական գործողությունները ծագում են ընդունված լավագույն փորձառություններից (գործելակարգից) և օգտագործվում են ծայրահեղ իրավիճակներում:

Ծրագրային ապահոման մշակման գործընթացում փոփոխությունն անխուսափելի է:
Սակայն միակ բանը, որ հնարավոր է որոշակի վստահությամբ կանխատեսել, անպայման փոփոխություն տեղի ունենալն է:

Ի հակադրություն հայտնի տեսակետի՝ ծրագրային ապահովման փոփոխության ծախսերը չեն աճում. նախագծման տարբեր փուլերում, փոփոխության հետ կապված, ծախսերի աճն աստիճանաբար նվազում է [2]: Վերջին տասնամյակների ընթացքում ամբողջ աշխարհում ծրագրային ապահովման մասնագետները հսկայական ռեսուրսներ ծախսելով, փորձել են նվազեցնել ծրագրային ապահովման նախագծերի փոփոխման ծախսերը, բայց հանգել են այն գաղափարին, որ անհրաժեշտ է կիրառել օբյեկտային կողմնորոշմամբ լավագույն ծրագրավոր-

ման լեզուները, տվյալների բազաների օբյեկտային կառուցվածքը, ծրագրավորման գործնական մոտեցումները, թեստավորման ժամանակակից մեթոդները և այլն: Այս ամենը հանգեցնում է ծայրահեղ ծրագրավորման տեխնիկական սկզբունքին: Եթե ժամանակի ընթացքում փոփոխման ծախսերը դանդաղ են աճում, ապա էապես փոխվում է որոշումների կայացման մոտեցումը: Այսպիսի տրամաբանությունը ենթադրում է, որ կարևոր որոշումներ կկայացնենք այնքան ուշ, որքան դա հնարավոր է, որպեսզի հետաձգենք որոշման կայացման հետ կապված ծախսերը [2]:

Այս ամենը նշանակում է, որ ճկունությունը ծայրահեղ ծրագրավորման կարևորագույն բնութագրերից է: Փոփոխությունը պայմանավորված է անորոշությամբ, իսկ վերջինիս հաղթահարման նպատակով մշակված ցանկացած գործընթացի հիմքում ընկած է ճկունությունը: Փոփոխության ընդունումը նշանակում է՝ անորոշությունը մեկնաբանել որպես անհրաժեշտ պայման, այլ ոչ թե նախագծի ստեղծման արգելք: Ընդգրկել փոփոխությունը՝ նշանակում է ընդգրկել ճկունությունը: Ծայրահեղ ծրագրավորման սկզբունքների և գործնական մոտեցումների մեծ մասը այս կամ այն կերպ կարող է հարաբերակցվել ճկունության հետ, որն անորոշության պայմաններում արժեք է ստեղծում: Որքան բարձր է անորոշության աստիճանը, այնքան մեծ արժեք է ստեղծում այն:

Այսպիսով՝ ճկունությունը հնարավոր է դիտարկել և մեկնաբանել որպես օպցիոն: Հետևաբար՝ ծայրահեղ ծրագրավորումը կարելի է ներկայացնել որպես օպցիոն, որը նախագծի ստեղծման գործընթացն ավելի շարժունակ է դարձնում: Ծայրահեղ ծրագրավորման նախագիծը ենթարկվում է գնահատման ընդհանուր հիմնական սկզբունքներին, ինչպես ցանկացած այլ իրական ակտիվ [3,4]:

Նմանատիպ ակտիվների գնահատման համար կօգտագործենք Բլեք-Շոուլսի ստանդարտ մոդելը: Այդ մոդելի հիմնական գաղափարը հենքային ակտիվի և այդ նույն դրամահոսքերով ոչ ռիսկային ակտիվի հիման վրա ճամպրուկի ստեղծումն է, և դրա համար էլ ունի նույն արժեքը, ինչը և գնահատվող

օպցիոնը: Քոլ օպցիոնի արժեքը Բլեք-Շոուլսի մոդելում կարելի է գրել որպես հինգ փոփոխականներով ֆունկցիա՝

S - հիմնական ակտիվի (ռիսկային) ընթացիկ արժեք,

K - օպցիոնի իրականացման արժեք,

t - օպցիոնի կյանքի տևողությունը՝ այն ժամանակահատվածը, որը մնացել է մինչև նրա լրացման պահը,

rf - ոչ ռիսկային տոկոսադրույքը, որը համապատասխանում է օպցիոնի կյանքի ժամկետին (տարեկան հաշվարկմամբ),

σ - հիմնական ակտիվի արժեքի փոփոխությունը ցույց տվող գործակցի բնական լոգարիթմի ցրվածքը (դիսպերսիան):

Քոլ օպցիոնի արժեքը՝ C-ն, կլինի՝

$$C = SN(d_1) - Ke^{-r_f t}N(d_2), \qquad (1)$$

որտեղ՝

$$d_1 = \frac{\ln\left(\frac{S}{K}\right) + \left(r_f + \frac{\sigma^2}{2}\right)t}{\sigma\sqrt{t}} \qquad (2)$$

$$d_2 = d_1 - \sigma\sqrt{t}\,. \qquad (3)$$

Մյուս ֆունկցիաներն ունեն հետևյալ բացատրությունները՝ $e^{-r_f t}$ ֆունկցիան ընթացիկ արժեքի գործոնն է և արտացոլում է այն փաստը, որ քոլ օպցիոնը պարտադիր չէ որ իրականացվի մինչև ժամկետի ավարտը: $N(d_1)$ և $N(d_2)$ ֆունկցիաները հավանականություններ են, որոնք գնահատված են նորմալ բաշխման ֆունկցիան օգտագործելու միջոցով:

σ պարամետրը սովորաբար անվանում են հիմնական ակտիվի փոփոխականություն, որն ամբողջական ռիսկի չափն է, այսինքն՝ շուկայական ռիսկի և կորպորատիվ ռիսկի գումարը: Նշված մեծությունը հաշվարկվում է որպես թվային շարքի ընտանդարտ շեղում, որն ստացվել է հիմնական ակտիվի եկամտաբերության գործակցի բնական լոգարիթմը վերցնելով որպես համապատասխան ժամանակահատվածներում հիմնական ակտիվի գների հարաբերակցություն [5]:

Այսպիսով՝ օպցիոնների տեսությունը հաստատում է, որ փոփոխման ծախսերի ավանդական մոդելի պայմաններում համակարգի բնութագրի վերաբերյալ որոշումները պետք է կայացվեն որքան հնարավոր է շուտ, քանի որ հապաղելն այդ իրավիճակում ցանկալի չէ:

Առանցքային բառեր. *ծրագրային ապահովում, կառավարում, ռիսկ, անորոշություն, ճկունություն:*

ԳՐԱԿԱՆՈՒԹՅՈՒՆ

1. **Гагарина Л.,** Кокорева Е., Виснадул Б. Технология разработки программного обеспечения. - М.: Форум, 2008. - 392с.
2. **Beck K.** Extreme Programming Explained: Embrace the Change (2nd Edition). - Addison Wesley, 2004. - 224p.
3. **Карслян Э.В.,** Петросян К.В. Модель инвестиций в научно-технические проекты в условиях существенных неопределенностей // Известия НАН РА и ГИУА. Сер. ТН. - 2010. - Том 63, N1. - С. 213-221.
4. **Карслян Э.В.,** Петросян К.В. Реальные опционы и гибкость менеджмента в контексте проектирования программного обеспечения // Информационные технологии и управление. 4-2. - Ереван, ,,Энциклопедия – Арменика”, 2006. - С. 8-13.
5. **Вайн С.** Опционы: Полный курс для профессионалов. 2-е изд. - М.: Альпина Бизнес Букс, 2007. - 465 с.

К. В. ПЕТРОСЯН, А. Г. ГРИГОРЯН

СРАВНИТЕЛЬНЫЙ АНАЛИЗ МЕТОДОВ УПРАВЛЕНИЯ ПРОЕКТАМИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Резюме

В статье рассматриваются основные понятия и определения, классификация программного обеспечения, этапы создания программного продукта в рамках жизненного цикла. Изложены не-

которые существующие подходы к оценке качества процессов создания программного обеспечения, произведен сравнительный анализ требований и спецификаций программного обеспечения, предлагается метод управления по созданию программмного обеспечения в условиях неопределенности. Для разработки этого метода применяется теория реальных опционов и методология экстремального программирования. Реальные опционы дают возможность разработчикам и менеджерам гибко изменять продукты и планы в условиях неопределенностей.

Ключевые слова: *программное обеспечение, управление, риск, неопределенность, гибкость.*

K. V. PETROSYAN, A. H. GRIGORYAN

COMPARATIVE ANALYSIS OF SOFTWARE PROJECT MANAGEMENT METHODS

Summary

The article deals with the basic concepts, definitions and classification of software development lifecycle. Some existing approaches of assessing the quality of the software development process were examined and a comparative analysis of requirements and definition of specifications for software was performed. As a result, a method is proposed to manage software development under uncertainty. For the development of the aforementioned method we turn to the theory of real options and extreme programming methodology, which provide developers and managers with valuable flexibility to change products and plans under uncertainty.

Key words: *software, management, risk, uncertainty, flexibility.*