20 Ordov K., Madiyarova A., Ermilov V., Tovma N., Murzagulova M. New trends in education as the aspect of digital technologies. International Journal of Mechanical Engineering and Technology, 2019, no 10(2), p. 1319-1330.

ТҮЙІН

COVID-19 короновирустық індетінің әлем бойынша таралуына байланысты пандемия жағдайы білім беру жүйесіне әсер етті. Қарастырылып отырған мақалада Қазақстан университеттеріндегі заманауи білім берудің басты ерекшеліктері, білім беруді цифрландырудың негізгі мәселелері, соның ішінде Жәңгір хан атындағы Батыс Қазақстан аграрлық-техникалық университетінің індетке байланысты күндізгі оқу нысанында оқитын білім алушылардың қашықтықтан оқытылуына көшу тәжірибесі қарастырылды.

Зерттеудің мақсаты білім беру ортасын өзгертуге байланысты заманауи білім беру үрдісін қамтамасыз ету үшін профессор-оқытушылар құрамының цифрлық сауаттылығын дамытудың негізгі құндылықтарын анықтау болып табылады. Зерттеудің өзектілігі қазіргі заманғы жоғары оқу орындарында білімнің ерекшелігін, әсіресе білім беру үрдісіне цифрлық технологияларды енгізу, білім беру жүйесін цифрландыру жағдайына бейімделу үшін оны дамытудың басты артықшылықтарын айқындау қажеттілігімен анықталады. Мақалада заманауи технологияларды жалпы қолдануда цифрлық құзыреттілік элементтерін анықтау тәсілі зерттеудің жаңалығы болып қарастырылады.

Ұсынылған мақалада қашықтықтан оқытуға арналған білім беру платформаларын қамтамасыздандыру мәселелері және online (онлайн) форматта дәріс өткізудің заманауи әдістеріне тоқталады. Сонымен қатар, білім беру саласында қолданылатын әр түрлі платформалар, олардың мақсаты мен міндеттері және қолданудың тиімді жақтары көрсетілген.

УДК 004.272.43
МРНТИ 50.33.04

Бекенова А.С., техника ғылымының магистрі, **негізгі автор,** https://orcid.org/0000-0002-2010-1488
«Жәңгір хан атындағы Батыс Қазақстан аграрлық-техникалық университеті» КеАҚ, Жәңгір хан көшесі, 51, Орал қ., 090009, Қазақстан Республикасы, inabat.77@mail.ru
Муталова Ж.С.,техника ғылымының магистрі, https://orcid.org/0000-0001-9912-5978
«Жәңгір хан атындағы Батыс Қазақстан аграрлық-техникалық университеті» КеАҚ, Жәңгір хан көшесі, 51, Орал қ., 090009, Қазақстан Республикасы, zhazira77@mail.ru
Бекенова С.С., техника ғылымының магистрі, https://orcid.org/0000-0001-7707-5623
Орал қаласы, sandu79@mail.ru

Bekenova A.S., master of technical sciences, **the main author,** https://orcid.org/0000-0002-2010-1488
NJSC «West Kazakhstan Agrarian and Technical University named after Zhangir khan», Uralsk, st. Zhangir khan 51, 090009, Kazakhstan, inabat.77@mail.ru
Mutalova Zh.S., master of technical sciences, https://orcid.org/0000-0001-9912-5978
NJSC «West Kazakhstan Agrarian and Technical University named after Zhangir khan», Uralsk, st. Zhangir khan 51, 090009, Kazakhstan, zhazira77@mail.ru
Bekenova S.S., master of technical sciences, https://orcid.org/0000-0001-7707-5623
Uralsk, sandu79@mail.ru

АҚПАРАТТЫҚ ЖҮЙЕЛЕРДІҢ МОНОЛИТТІ ЖӘНЕ МИКРОСЕРВИСТІ СӘУЛЕТІН САЛЫСТЫРМАЛЫ ТАЛДАУ
COMPARATIVE ANALYSIS OF MONOLITHIC AND MICROSERVICE ARCHITECTURES OF INFORMATION SYSTEMS

Аннотация

Қазіргі уақытта, Netflix, Apple, Instagram және Pinterest сияқты көптеген компаниялар өздерінің бағдарламалары мен жүйелерін микросервистерге ауыстыруда, , өйткені бұл сәулет компанияларға сәйкес есептеу ресурстарын масштабтауға, оларды пайдалануға мүмкіндік береді.

Заманауи веб-қосымшалардың толыққанды жұмыс істеуі үшін, бағдарламалық интерфейсті ұсыну мүмкіндігі сияқты жұмыстар, көптеген сұраныстарды өңдеу, масштабтау, қамтамасыз ету, деректерге қол жеткізудің жоғары жылдамдығы, жоғары сенімділікті қамтамасыз ету, үзіліссіз орнықты жұмыс істеу сияқты жоғары талаптар болуы керек. Аталған алыптардың әртүрлі мәселелері микросервистерге көшу арқылы шешілген. Бұл компаниялар икемді микросервистерді инновацияларды енгізу және шығыдарды оңтайландыру үшін пайдаланудың жақсы мысалы болып табылады. Дүние жүзіндегі мыңдаған басқа брендтер де осылай микросервисті сәулеттің жаңа мүмкіндіктерін қолдануда.

Бұл мақалада масштабталатын веб архитектурасы ұғымы, оның түрлері, жобалау принциптері және инфрақұрылымдық талаптар зерттелді. Ақпараттық жүйелердің монолиттен микросервистік сәулетке көшу мәселесі егжей-тегжейлі талқыланып, күрделілігі мен ерекшелігі талданды. Сәулеттің әр түрлі типтеріне - монолитті, SOA және микросервис түрлеріне, олардың әлсіз және күшті жақтары, жүйені жобалаудың негізгі өлшемдерін ескере отырып, жоба және команда күйі негізінде салыстырмалы талдау жасалынған. Мақалада ең алдымен екі сәулетті салыстырмалы бағалау бойынша жүргізілген нәтижелерді жалпылауға бағытталған. Екіншіден, ол монолитті құрылымнан микросервистікке ауысатын процессор, жадты тұтыну, желінің өнімділігі, дискілік операциялар және даму уақыты, интеграция сияқты әртүрлі айнымалылар арасындағы өнімділік пен өзара байланысты анықтауға бағытталған.

ANNOTATION

Currently, many companies, such as Netflix, Apple , Instagram, and Pinterest, are replacing their programs and systems with microservices, as this architecture companies to scale their computing resources accordingly, using them.

For the full functioning of modern web applications, it is necessary to have high requirements, such as the ability to provide a software interface, processing, scaling, providing a large number of requests, high data access speed, ensuring high reliability, stable operation without interruption. Various problems of these giants were solved by switching to microservices. This is a good example of how companies use flexible microservices to innovate and optimize costs. Thousands of other brands around the world are also using new features of microservice architecture.

This article examines the concept of scalable web architecture, its types, design principles, and infrastructure requirements. The problem of the transition of information systems from monolithic to microservice architecture was discussed in detail, the complexity and specifics were analyzed. A comparative analysis is carried out on the basis of various types of architecture - monolithic, SOA and microservice types, their weaknesses and strengths, project and team status, taking into account the main criteria for designing the system. The article is primarily aimed at generalizing the results of comparative assessments of two architectures. Secondly, it aims to determine the performance and interconnection between various variables such as processor, memory consumption, network performance, disk operations and development time, integration, which are moving from a monolithic structure to microservices.

Кілтті сөздер: *монолитті сәулет, микросервистік сәулет, өнімділік, масштабталу, сенімділік, іске асырудың күрделілігі, веб-қосымша, ақпараттық жүйе.*

Key words:. *monolithic architecture, microservice architecture, performance, scalability, reliability, complexity of implementation, web application, information system.*

Кіріспе. Қазіргі уақытта электронды құрылғылар санының айтарлықтай өсуі байқалады

Интернетке шығу мүмкіндігі бар құрылғылардың бұл өсуі оларды масштабтауға байланысты , веб-қызметтерге жүктемені арттыру сияқты әртүрлі мәселелер туындатты.

Осы мәселелерге байланысты бағдарламашылар олардың жүйелерін талаптарға бейімдей отырып, сәулетті дамытуға көбірек уақыт бөлуге мәжбүр.

Ұйымдарға технологияға негізделген сенімді шешімдер қажет. Сондықтан бағдарламалық жасақтама жасаушылар уақыт өте келе бағдарламалық жасақтама өнімдеріне ресурстық тиімді және функционалдық талаптарға сәйкес болуға көмектесетін сәулеттің әртүрлі түрлерін әзірледі және енгізді. Олардың модульдері бір деңгейде немесе әртүрлі деңгейлер бойынша таратылады [1].

Бағдарламалық жүйені, оның барлық коды бір түйінде жалғыз процесс ретінде орналастырылып және іске қосылатын болса, монолитті деп атауға болады. Монолитті сәулетке сәйкес салынған жүйенің типтік мысалы қарапайым веб-қосымшасы болып табылады. Жалпы жағдайда, мұндай веб-қосымша - инфрақұрылымдық көзқарас бойынша клиент сұрауларын қабылдайтын және өңдейтін, сонымен қатар, қажет болғанда жүйенің деректер қорына жүгінетін машина.[2] Жүктеме өскен сайын жүйеде пайдаланушы әрекеттеріне жауап беру уақыты айтарлықтай ұзақ болуы мүмкін, себебі, қабылдау, өңдеу, деректер қорына сұраныс бір машинада бір процесспен жұмыс істейді.

Монолитті сәулет, бағдарламалық қамтамасыз ету өндірісінде дәстүрлі түрде виртуалды машиналармен бірге пайдаланылады. Бағдарламалық жүйелер пайда болғаннан бері шағын және ауқымды жобалар үшін сәтті және тиімді формула болды.

Монолитті қосымшалардың өнімділігі көлемді болған кезде төмендейтіні белгілі. Өңделген деректер өткізу қабілеттілігінің белгілі бір деңгейінен асады немесе асып түседі.
Уақыт өте келе әртүрлі шешімдер пайда болды. Мысалы, жаңа технологияларға көшу, тәуелсіз қызметтерді басқару және қуатты серверлер т.б. Шешімнің дұрыс таңдалмауы (яғни бағдарламалық жасақтамадағы ақаулар, модульділік, үйлесімділік және байланыс конфигурациясының сәйкессіздігі, сондай-ақ қауіпсіздік деңгейінің жеткіліксіздігі) болашақта ресурстардың жоғары шығындарына әкелуі мүмкін. Соңғы екі онжылдықта оңтайлы шешімдерді ұсынатын инновация жаңа сәулеттердің пайда болуына әкелді. Бұл тұрғыда микросервистердің сәулеті қарқын алуда және ол техникалық, қаржы және маркетинг салаларында шешім қабылдау процесінің бөлігі болып табылады. Микросервистер монолитті сәулеті бар, таратылған жүйеге бағытталған және оқшауланған қызметі бар жүйелерді ауыстырады [3].

Микросервистерді қолдану кезінде туындайтын мәселелердің бірі оны жүзеге асыру үшін қажет күш-жігермен байланысты және бұлттағы әрбір микросервисті масштабтау. Сонымен қатар, микросервистерді орналастыратын компаниялар DevOps, Docker, Chef, Puppet және масштабтау автоматты сияқты әртүрлі автоматтандыру құралдарын қолдана алады. Мұндай құралдарды енгізу уақыт пен ресурстарды үнемдейді. Өкінішке орай, бұл одан әрі даму, көші-қон және интеграция үшін қажет. Сондықтан ептілікке, дербес өсуге,және ауқымдылық, инфрақұрылымдық шығындар үлгілері аталғандарды қабылдайтын компаниялар үшін басты мәселе болып табылады[4].
Тағы бір мәселе - өндірісте микросервистерді ұйымдастыру. Оның үстіне, микросервистердің шешімін беретін техникалық қиындықтар ұйымдастырушылық қиындықтарға әкеледі. Ұйым пайда болған техникалық мәселелердің көпшілігін немесе барлығын шеше алатын болса да, ұйымның құрылымы мен дағдылары да жаңа архитектурамен үйлесімді болуы қажет [5]. Әртүрлі шешімдер болса да бұрыннан бар монолитті сәулеттен микросервистерге көшу процесі әлі де дәлдікке ие емес. Сонымен қатар, қысқаша әдебиеттерге шолу бұл сәулеттер арасындағы айырмашылықтарды ашты [6].

Зерттеу материалдары мен әдістері. Осы мәселелерге сүйене отырып, бұл зерттеу ең алдымен екі сәулетті салыстырмалы бағалау бойынша жүргізілген нәтижелерді жалпылауға бағытталған. Екіншіден, ол монолитті құрылымнан микросервистікке ауысатын процессор, жадты тұтыну, желінің өнімділігі, дискілік операциялар және даму уақыты, интеграция сияқты әртүрлі айнымалылар арасындағы өнімділік пен өзара байланысты анықтауға бағытталған. Осы мәселемен жұмыстану барысында талдау, салыстырмалы талдау, аналитикалық зерттеулер әдістері қолданылды.

Нәтижелер және оларды талдау. Қосымшаларды әзірлеу мен орналастырудың ең жақсы әдісін түсіну бүгінгі таңда деректерді басқаратын кез-келген ұйым үшін маңызды фактор болып табылады.

Қызметке бағытталған сәулет (SOA) [7] және микросервистер сияқты опциялар дәстүрлі монолитті тәсілдерге сәйкес келмейтін қосымшаларды құру және іске қосу үшін құнды

икемділікті ұсынады. Дегенмен, қайсысы жақсы екенін анықтау үшін олардың арасындағы айырмашылықтарды түсінген жөн.

Біз сәулеттің әр түріне салыстырмалы талдау жүргіздік. Бағалаудың негізгі критерийлері: жобалау, ауқымдылығы, икемділік және әзірлеу. Талдау деректері 1-кестеде келтірілген.

Қазіргі уақытта Netflix, Amazon және eBay сияқты көптеген компаниялар, өз қосымшалары мен жүйелерін ауыстырып, бұлтты пайдаланады. Микросервисті сәулет бұлттық есептеу моделі болғандықтан, олардың қолданылуына сәйкес бұл компанияларға есептеу ресурстарын кеңейтуге мүмкіндік береді. [8].

Кесте 1 – Сәулеттің әртүрлі түрлерін салыстырмалы талдау

Бағалау критерийлері	Микросервистер	SOA	Монолит
Жобалау	Микросервистер бір-бірімен өзара әрекеттесетін шағын қосымшалар түрінде тұрғызылған	Сервистердің мөлшері әр түрлі шағын қосымшалардан бизнес функцияларды қамтитын өте ірі корпоративтік сервистерге дейін	Монолитті қосымшалар орасан үлкен мөлшерге дейін өсе алады, мұндай жағдайларда қосымшалардың толықтығын түсіну қиын.
Ауқымдылығы	Микросервистер бір-бірінен тәуелсіз сервистерден құралады.	Қызметтер арасындағы тәуелділік пен қайталанып қолданылатын компоненттер арасындағы ауқымдылық мәселесі туындайды.	Монолитті қосымшалар ауқымдылығынан мәселелер жиі туындайды.
Икемділігі	Қосымшалардың жоғары дәрежедегі икемділігін қамтамасыз ете отырып, кішкентай тәуелсіз өрістетілетін модульдер жинақтау мен шығаруды басқаруды қысқартады.	Компоненттерді бірігіп қолданудың жоғары дәрежесі тәуелділікті арттырады және басқару мүмкіндігін шектейді.	Қосымшалардың монолитті артефактілерін қайтадан енгізу кезінде икемділік алу қиындайды.
Әзірлеу (өңдеу)	Микросервистерді әзірлеу дайындаушыларға қойылған мақсатқа сәйкес келетін ортаны жасауға мүмкіндік береді.	Көпретті компоненттер және стандартты әдістер әзірлеушілерге іске асыру кезінде көмектеседі.	Монолитті қосымшалар әзірлеу кезіндегі басқа технологиялардың енуін шектейтін өңдеудің бір стегін қолданумен жүзеге асады.

Жүйелер немесе қосымшаларға монолитті немесе микросервисті сәулетті таңдаған кезде контекстті дұрыс түсіну керек. Кейбіреулер микросервистерден сенімді түрде бірден бастайды, ал басқа жүйелер басында монолитті таңдап, олардың стартаптары өскен сайын, сайып келгенде микросервистерге көшеді. Жүйеге қандай сәулет сәйкес келетінін анықтау үшін оның контекстісін және сценарийлерді қарастыру керек. [9].

Монолитті сәулет қосымшаларды құрудың дәстүрлі әдісі болып саналады. Монолитті қосымша біреу және бөлінбейтін тұтас болады. Әдетте мұндай шешім клиенттің интерфейсін, серверлік қосымшаны және дерекқорды қамтиды. Ол бірыңғай және барлық мүмкіндіктер бір жерде басқарылады және қызмет көрсетіледі. Әдетте монолитті қосымшаларда бір үлкен код базасы бар және онда модульділігі жоқ. Егер әзірлеушілер бірдеңе жаңартқысы келсе немесе өңдеу үшін олар сол код базасына қол жеткізе алады. Осылайша, олар бір уақытта бүкіл стекке өзгерістер енгізеді.[10].

Монолитті сәулеттің күшті жақтары:

✓ Жанама мәселелер аз. Жанама мәселелер - бұл барлық қосымшаларға әсер ететін мәселер, кэштеу және өнімділікті бақылау. Монолитті қолданбаның бұл функционалдылық аймағы тек біреуіне қатысты, сондықтан онымен жұмыс істеу оңайырақ.

✓ Қарапайым жөндеу және тестілеу. Микросервистік сәулеттен айырмашылығы монолитті қосымшаларды тестілеу әлдеқайда қарапайым, себебі монолитті қосымша бұл бір бөлінбейтін блок.

✓ Орналастыру қарапайымдылығы. Монолитті қосымшалар туралы сөз болғанда, сізге көп орналастыруды өңдеудің қажеті жоқ - тек біреуі файл немесе каталог.

✓ Әзірлеу оңайлығы. Әзірге монолитті тәсіл қосымшаларды құрудың стандартты әдісі, кез келген инженерлік топтың монолитті қосымшаны әзірлеуде қажетті білімі мен мүмкіндіктері бар. [11].

Монолитті сәулеттің әлсіз жақтары:

✓ Код пен құрылымды түсіну. Монолитті сәулет кеңейген кезде кодтарды түсіну өте қиын болады. Сонымен қатар, бір қосымшадағы күрделі код жүйесін басқаруда күрделі болады.

✓ Өзгерістер енгізу. Өте күшті тәуелділігі бар күрделі жүйелерде үлкен өзгерістерді жүзеге асыру қиынырақ. Кез келген кодты өзгерту бүкіл жүйеге әсер етеді, сондықтан ол мұқият үйлестірілген. Бұл жалпы даму процесін әлдеқайда ұзағырақ жасайды.

Ауқымдылығы. Компоненттерді масштабтау мүмкін емес, тек бүкіл бағдарлама масштабталады. [12].

✓ Жаңа технологияларды қолдануға кедергі. Монолитті сәулетте жаңа технологияны қолдану мәселелі, себебі содан кейін барлық бағдарлама қайтадан жазылу керек.

Монолитті қосымша біртұтас тұтастық болып табылады, микросервис архитектурасы оны кіші тәуелсіз бірліктер жиынтығына бөледі. Бұл бөлімшелер жеке қызмет ретінде өтініш беру процесін әрқайсысы орындайды. Осылайша, барлық қызметтер өздерінің логикасы мен дерекқоры, сонымен қатар белгілі бір функциялары бар бөліктерден тұрады. [13].

Мартин Фоулер микросервистің сәулетін бір қосымша ретінде жұмыс істейтін шағын қызметтер жиынтығын әзірлеу тәсілі ретінде анықтады. Қызметтер HTTP ресурстары, API сияқты жеңіл механизмдер арқылы байланысады және әр қызмет өз бетінше жұмыс істейді [12].

Басқаша айтқанда, микросервис сәулетінде барлық функционалдылық өзара әрекеттесетін дербес орналастырылатын модульдерге бөлінген API деп аталатын белгілі бір әдістер арқылы бір-бірімен байланысқан [13]. Әрбір сервис жеке аймақта жаңартылатын, орналастырылған және тәуелсіз масштабталатын өз қызметін қамтиды. [14].

Микросервисті сәулеттің күшті жақтары[15].*:*

✓ Тәуелсіз компоненттер
✓ Басқару мен түсінудегі қарапайымдылық
✓ Жақсырақ масштабталу
✓ Технологияны таңдаудағы икемділік
✓ Ақауларға төзімділіктің жоғары деңгейі

Микросервисті сәулеттің әлсіз жақтары:

✓ Қосымша күрделілік
✓ Таратылған жүйе
✓ Микросервистік қосымшаны құру кезінде барлығына әсер ететін тіркеу, кэштеу және өнімділік мониторингі сияқты бірқатар мәселелерге тап боласыз. [16].
✓ Тестілеу [17].

2-кестеде монолитті сәулет пен микросервисті сәулетті жүйені жобалаудың негізгі критерийлері бойынша салыстыру жүргізілді.

Кесте 2 – Монолитті сәулет пен микросервисті сәулетті жүйені жобалаудың негізгі критерийлері бойынша салыстырмалы талдау

Критерий	Монолит	Микросервис
Өрістету (Deployment)	Бір рет орналастыруға мүмкіндік береді, содан кейін ағымдағы өзгерістер негізінде баптай аласыз. Егер бірдеңе дұрыс болмаса, жобаның бәрі бұзылады.	Көп қосымша жұмыс жасай алады, әр қызмет тәуелсіз болады. Бірақ үлкен артықшылығы бар, егер ақау болса, тек бір кішкентай микросервисті бұза аласыз, бұл барлық жүйені бұзуға қарағанда тиімдірек.
Қызмет көрсету (Maintenance)	Бұл біртұтас қосымша болғандықтан қызмет көрсету өте қарапайым	қозғалмалы бөлшектер жиынының көмегімен басқарылатын инфрақұрылым, DevOps білімі қажет және контейнерлермен (Docker) және оркестрмен (Kubernetes, Docker Swarm), жұмыс істей білуі керек[18].
Сенімділік (Reliability)	Кез келген бұзылу, тіпті кішкентай, функционалдық бүкіл жүйенің істен шығуына әкеп соқтырады	Бір микросервистің істен шығуы барлық жүйенің істен шығуына әкелмейді
Ауқымдылығы (Scalability)	Қиын масштабтау, Көлденең масштабтау қолданылмайды.	Жақсы масштабтау[19].
Құны (Cost)	Үлкен емес қосымшалар үшін арзандау хостел қоса аласыз, ал үлкен қосымша үшін бағасы өте қымбатқа түседі.	Ең жақсы ауқымдылығы негізінде автоматты масштабтау және тек оны пайдаланушылар көлемді болған кезде төлей аласыз. Сонымен қатар, жұмыс күйіндегі инфрақұрылымды қолдау үшін ақысы төленген құрылғылар қажет.
Әзірлеу (Development)	Қосымша жылдам қосылады, бірақ күн өткен сайын код тозады да, жобаны әрі қарай әзірлеу қиын болады.	Басында контейнерлеу және оркестрация сияқты қосымша заттар әзірлеу және интеграция қиын болады, уақыт өте келе әзірлеу түсінікті әрі икемді болады.
Шығарылымдар (Releasing)	Бөлуге болмайтын ішкі тәуелділіктер болады. Командадағы өзгерістерге қатты тәуелді боласыз.	Жаңа функцияларды жылдам, жиі әрі сапалы шығаруға мүмкіндік береді.

Тағы бір зерттеу жүргізілді, оның барысында бірнеше сценарийлер алынды, олар қандай түрді көрсетеді, сәулет қай кезеңге байланысты қолданылуы керек, сіздің жүйеңіз не талап

етеді деген сұрақтарға жауап береді. [20]. Алынған сценарийлері бар деректер 3 – кестеде бейнеленген.

Кесте 3 – Монолитті және микросервисті сәулеттің өнім мен командалар күйлерінің негізгі сценарийлерінің базасында салыстырмалы талдау

Фактор	Монолитті қолдану	Микросервисті қолдану
Команда	Сіздің командаңыз бастапқы стадияда. Командада 2-5 адам бар. Жоғары деңгейдегі микросервисті сәулетпен жұмыс істей алмайды.	Командада оннан артық әзірлеуші бар. Әзірлеудің әр түрлі бағытындағы терең білімді қажет етеді.
Жоба стадиясы	Тексерілмеген жобаны жасайсыз және өмірге қабілеттілігі тексеріледі.	Қолданылу саласында терең экспертиза бар және ұзақ уақыт жобамен айналысасыз. Сіздің өніміңіз кең қолданылушылық аудиторияны қамтиды.
Қызметті жеткізу	Сіз өнімді бір рет өзіңізге қолайлы уақытта жаңарта аласыз.	Өнімнің бір бөлігі бір айда бір рет, қалғандары күнара жаңартылады.
Өнімділік	Сізге жоғары өнімділік қажет емес, себебі сіз аз трафикпен жұмыс жасайсыз.	Жүйенің кейбір бөлігі өспелі трафиктермен және жүктемелермен жұмыс жасайтын жоғары өнімділікті қажет етеді,

Қорытындылай келе, сәулетті оған қойылатын талаптарға сүйене отырып таңдау және құру керек екенін атап өткен жөн. Егер жоспарлау кезеңіндегі жүйе өзінің өмірлік циклі ішінде кішкентай болса, содан кейін оны микросервисті сәулет бойынша салу практикалық емес, өйткені оны жүзеге асыру әлдеқайда қиын және үлкен еңбек шығындарын талап етеді. Егер жүйені кеңейту жоспарланбаған болса немесе көлденеңінен масштабтау үшін дұрыс шешім монолитті стильде әзірлеу болады.

Микросервисті сәулет үлкен немесе өте үлкен жүйелердің мәселелерін шешуге арналған. Егер сіз микросервисті сәулетте жүйені салсаңыз, микросервисті сәулеттің негізгі артықшылықтарымен әзірлеушілер өнімді қолдау кезеңінде кездеседі. Жүйені масштабтау, ақауларды жою, кеңейту оңай. [20].

Осы деректердің негізіндегі салыстырмалы талдау нәтижесінде микросервистерді қолдану тиімділігі байқалады. Оның басқа сәулетпен салыстырғандағы артықшылығы - микросервистер өзара тәуелсіз жұмыс жасайтын жеке сервистерге бөлінгендігі.

ӘДЕБИЕТТЕР ТІЗІМІ

1 Карминский А.М. Методология создания информационных систем; ИНФРА-М, 2018. - 282 **с.**

2 Бен Фаррелл Веб-компоненты в действии; ДМК, 2020.-463 с.

3 Ньюмен Сэм. От монолита к микросервисам, Издательство: БХВ-Петербург: 2021. - 274 с.

4 Сэм Ньюмен Создание микросервисов, Издательство:Питер:2016. – 304 с.

5 Florian Auer From monolithic systems to Microservices: An assessment framework: Information and Software Technology 137 (2021) 106600

6 Парминдер Синг Кочер Микросервисы и контейнеры Docker, Издательство:ДМК - Москва: 2019.- 242с.

7 Thomas Erl Service-Oriented Architecture: Analysis and Design for Services and Microservices, Издательство:Pearson Service Technology, 2019.-393 с.

8 S. Hussain, J. Keung, and A. A. Khan, (2017). “Software design patterns classification and selection using text categorization approach”. Applied Soft Computing Journal, vol.58, pp. 225–244. http://doi.org/10.1016/j.asoc.2017.04.043

9 Кристиан, Хорсдал Микросервисы на платформе. NET / Хорсдал Кристиан. - М.: Питер, 2018. - 442 с.

10 S. Orlov and A. Vishnyakov (2017). “Decision Making for the Software Architecture Structure Based on the Criteria Importance Theory”, Procedia Computer Science, Vol 104, pp. 27-34, ISSN 1877-0509.

11 N. Bessis, X. Zhai, S. Sotiriadis, Service-oriented system engineering, Future Gener. Comput. Syst., vol. 80, pp. 211–214, Mar. 2018.

12 Мартин Фаулер Рефакторинг кода на JavaScript: улучшение проекта существующего кода, Диалектика, 2022, 466 с.

13 Chris Richardson Microservices patterns, Москва, 2018 – 520 p.

14 D. Taibi, K. Systä, A decomposition and metric-based evaluation framework for microservices, in: Cloud Computing and Services Science, 2020, pp. 133–149.

15 A. de Camargo, I. Salvadori, R.d.S. Mello, F. Siqueira, An architecture to automate performance tests on microservices, in: International Conference on Information Integration and Web-Based Applications and Services, 2016, 422–429 pp.

16 Dragoni, N., Giallorenzo, S. Microservices: Yesterday, Today, and Tomorrow. Present and Ulterior Software Engineering, 2017, 195-216pp.

17 Эпп В.В. Качество и тестирование программного обеспечения. Учебное пособие.// Пенза: Изддательство: ПГУ, 2016.-129 с.

18 Гвоздева В. А., Лаврентьева И. Ю. Основы построения автоматизированных информационных систем; синтег - Москва, 2016**.** - 320 с.

19 Водяхо А.И., Выговский Л.С., Архитектурные решения информационных систем: учебник. – Лань, 2017.-356 с.

20 Советов Б.Я. Архитектура информационных систем; Академия (Academia) - М., 2017**.** - 934 с.

REFERENCES

1 Karminskij A.M. Metodologiya sozdaniya informacionnyh sistem; INFRA-M,2018.-282s.

2 Ben Farrell Veb-komponenty v dejstvii; DMK, 2020.-463 s.

3 N'yumen Sem. Ot monolita k mikroservisam, Izdatel'stvo: BHV-Peterburg: 2021. - 274 s.

4 Sem N'yumen Sozdanie mikroservisov, Izdatel'stvo:Piter:2016. – 304 s.

5 Florian Auer From monolithic systems to Microservices: An assessment framework: Information and Software Technology 137 (2021) 106600

6 Parminder Sing Kocher Mikroservisy i kontejnery Docker, Izdatel'stvo:DMK - Moskva: 2019.- 242s.

7 Thomas Erl Service-Oriented Architecture: Analysis and Design for Services and Microservices, Izdatel'stvo:Pearson Service Technology, 2019.-393 s.

8 S. Hussain, J. Keung, and A. A. Khan, (2017). “Software design patterns classification and selection using text categorization approach”. Applied Soft Computing Journal, vol.58, pp. 225–244. http://doi.org/10.1016/j.asoc.2017.04.043

9 Kristian, Horsdal Mikroservisy na platforme. NET / Horsdal Kristian. - M.: Piter, 2018. - 442 s.

10 S. Orlov and A. Vishnyakov (2017). “Decision Making for the Software Architecture Structure Based on the Criteria Importance Theory”, Procedia Computer Science, Vol 104, pp. 27-34, ISSN 1877-0509.

11 N. Bessis, X. Zhai, S. Sotiriadis, Service-oriented system engineering, Future Gener. Comput. Syst., vol. 80, pp. 211–214, Mar. 2018.

12 Martin Fauler Refaktoring koda na JavaScript: uluchshenie proekta sushchestvuyushchego koda, Dialektika, 2022, 466 s.

13 Chris Richardson Microservices patterns, Москва, 2018 – 520 p.

14 D. Taibi, K. Systä, A decomposition and metric-based evaluation framework for microservices, in: Cloud Computing and Services Science, 2020, pp. 133–149.

15 A. de Camargo, I. Salvadori, R.d.S. Mello, F. Siqueira, An architecture to automate performance tests on microservices, in: International Conference on Information Integration and Web-Based Applications and Services, 2016, 422–429 pp.

16 Dragoni, N., Giallorenzo, S. Microservices: Yesterday, Today, and Tomorrow. Present and Ulterior Software Engineering, 2017, 195-216pp.

17 Epp V.V. Kachestvo i testirovanie programmnogo obespecheniya. Uchebnoe posobie.// Penza: Izddatel'stvo: PGU, 2016.-129 s.

18 Gvozdeva V. A., Lavrent'eva I. YU. Osnovy postroeniya avtomatizirovannyh informacionnyh sistem; sinteg - Moskva, 2016. - 320 s.

19 Vodyaho A.I., Vygovskii L.S., Arhitekturnye resheniya informacionnyh sistem: uchebnik. – Lan', 2017.-356 s.

20 Sovetov B.YA. Arhitektura informacionnyh sistem; Akademiya (Academia) - M., 2017. - 934 s.

РЕЗЮМЕ

В настоящее время многие компании, такие как Netflix, Apple, Instagram и Pinterest, меняют свои программы и системы на микросервисы, поскольку эта архитектура позволяет компаниям масштабировать вычислительные ресурсы в соответствии с ними, используя их.

Для полноценной работы современных веб-приложений должны быть такие работы, как возможность предоставления интерфейса программного обеспечения, обработка большого количества запросов, масштабирование, обеспечение, высокая скорость доступа к данным, обеспечение высокой надежности, стабильная бесперебойная работа. Различные проблемы указанных гигантов решались путем перехода на микросервисы. Эти компании являются хорошим примером использования гибких микросервисов для внедрения инноваций и оптимизации издержек. Тысячи других брендов по всему миру также используют новые возможности микросервисной архитектуры.

В данной статье изучено понятие масштабируемой веб-архитектуры, ее виды, принципы проектирования и инфраструктурные требования. Подробно обсуждался вопрос перехода информационных систем от монолита к микросервисной архитектуре, анализировались сложность и специфика. Проведен сравнительный анализ различных типов архитектуры - монолитных, SOA и типов микросервисов, их слабых и сильных сторон, состояния проекта и команды с учетом основных критериев проектирования системы. Статья направлена прежде всего на обобщение результатов, проведенных по сравнительной оценке двух архитектур. Во-вторых, он фокусируется на определении производительности и взаимосвязей между различными переменными, такими как процессор, потребление памяти, производительность сети, дисковые операции и время разработки, интеграция, которая переходит от монолитной структуры к микросервисной.

УДК 004.021
МРНТИ 20.23.19

Жаксыбаев Д. О., магистр педагогических наук, **основной автор,** https://orcid.org/0000-0001-6355-5431
НАО «Западно-Казахстанский аграрно-технический университет имени Жангир хана», г. Уральск, ул. Жангир хана 51, 090009, Казахстан, darhan.03.92@mail.ru

Zhaxybayev D. O., Master of Pedagogical Sciences, **the main author,** https://orcid.org/0000-0001-6355-5431
NJSC «West Kazakhstan Agrarian and Technical University named after Zhangir khan», Uralsk, st. Zhangir khan 51, 090009, Kazakhstan, darhan.03.92@mail.ru

АЛГОРИТМЫ ПОИСКА ИНФОРМАЦИОННЫХ РЕСУРСОВ ДЛЯ ПОВЫШЕНИЯ ЭФФЕКТИВНОСТИ РАБОТЫ С ДОКУМЕНТАМИ
ALGORITHMS FOR FINDING INFORMATION RESOURCES TO IMPROVE THE EFFICIENCY OF PAPERWORK MANAGEMENT

Аннотация

В данной статье описывается основа хранения, восстановления и обновления информации с особым акцентом на алгоритм поиска и структуру данных, необходимую для