УДК 004.273

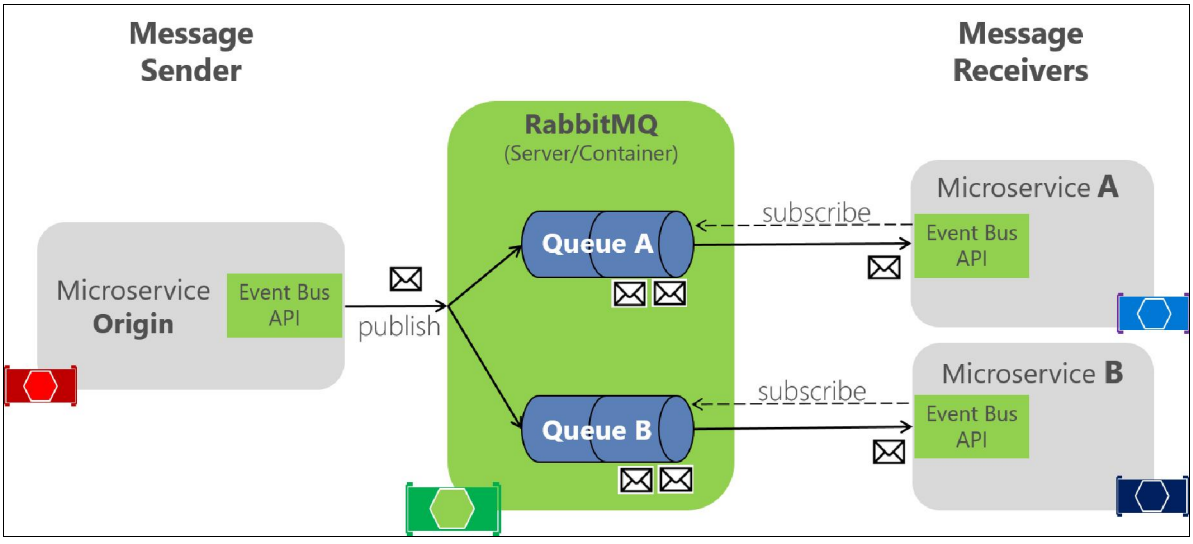
ПЕРАВЫКАНАННЕ НЕАПРАЦАВАНЫХ ПАВЕДАМЛЕННЯЎ У СІСТЭМАХ МІКРАСЕРВІСНАЙ АРХІТЭКТУРЫ З ВЫКАРЫСТАННЕМ БРОКЕРА RABBITMQ

Будкоўскі Г.Л.

Навуковы кіраўнік – Цярноў Я.В., к.т.н.

Згодна з сучаснымі тэхналогіямі распрацоўкі праграмнага забяспячэння

(ПЗ) пры пабудове размеркаваных інфармацыйных сістэм перавага часцей аддаецца мікрасервіснай, чым маналітнай архітэктуры. Як правіла, гэта адбываецца па прычыне лепшай прыстасаванасці сістэмы мікрасервіснай архітэктуры да наступнага маштабавання і павышанай надзейнасці яе работы, калі непрацаздольнасць асобнай часткі не прыводзіць да непрацаздольнасці ўсёй сістэмы [1]. Для правільнай работы сістэмы патрабуецца ўзгадненне зместу некалькіх баз дадзеных, размеркаваных па мікрасервісах. Змест названых баз дадзеных узгадняецца перасылкай паведамленняў паміж мікрасервісамі пры дапамозе адмысловага ПЗ – брокера, альбо пасярэдніка паведамленняў [2] (малюнак 1).



Малюнак1. Прыклад пабудовы мікрасервіснай архітэктуры з выкарыстаннем брокера паведамленняў RabbitMQ [2]

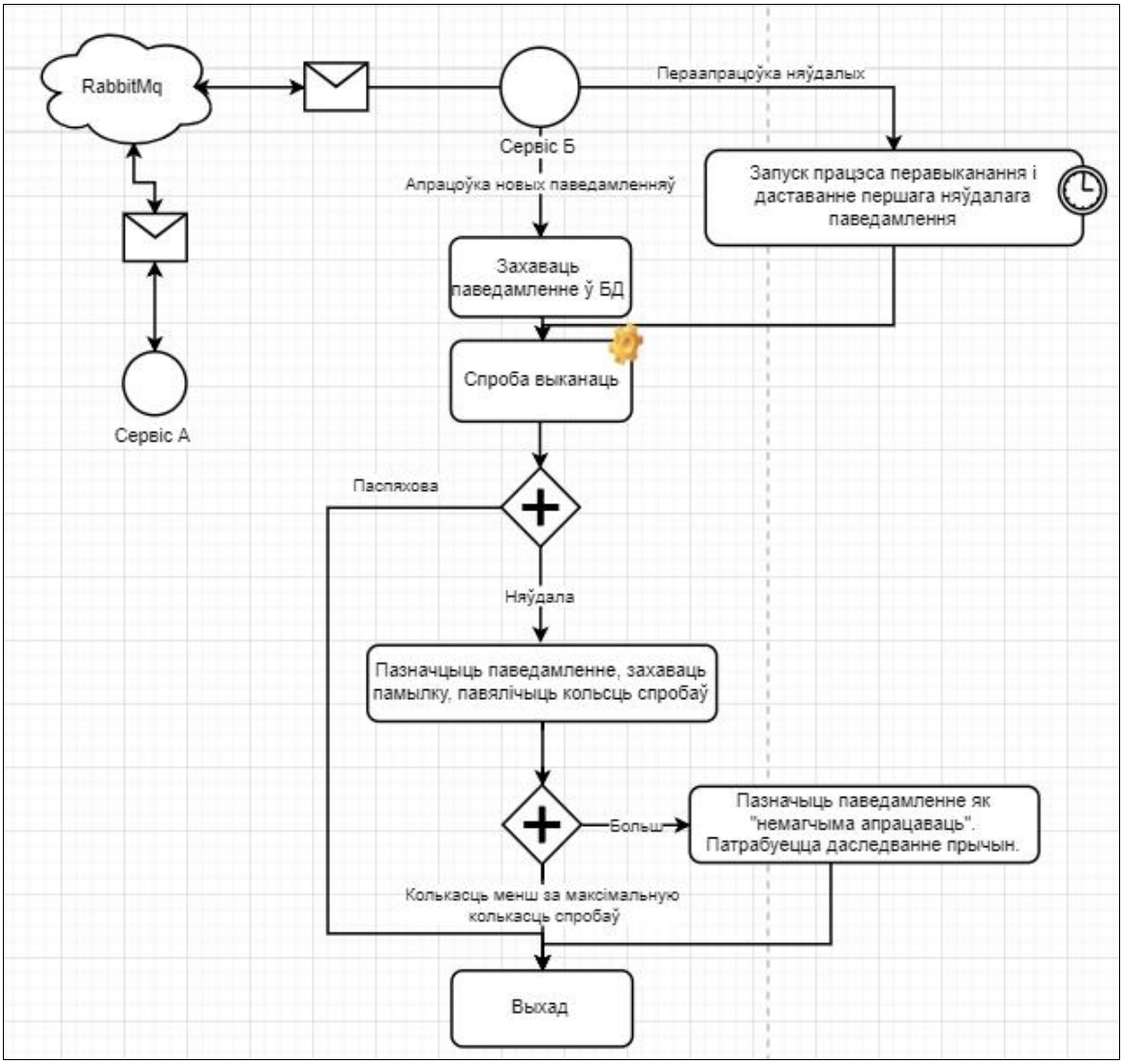
Пры распрацоўцы ПЗ часта аддаецца перавага брокеру паведамленняў RabbitMQ за наступныя станоўчыя якасці [3]:

- бясплатнасць і адкрыты зыходны код;
- нязначны аб’ём памяці падчас працы (меньш за 40 Мбайт);
- незалежнасць ад платформы і яе вытворцы;
- магчымасць стварэння асобных мікрасервісаў на розных мовах праграмавання;
- некалькі ўзроўняў бяспекі пры перадачы дадзеных і інш.

Разам з тым брокер RabbitMQ не забяспечвае перавыкананне спажывецкіх паведамленняў у выпадку памылак падчас іх апрацоўкі. Паведамленне можа апрацавацца няўдала з наступных прычын:

- адначасовая спроба змены адных і тых дадзеных з боку рознага ПЗ;
- затрымка выканання чаргі паведамленняў;
- часовая адсутнасць дадзеных, звязаных з паведамленнем, і інш.

Праграміст павінен самастойна клапаціцца пра перавыкананне спажывецкага паведамлення. Для вырашэння названай праблемы аўтарам прапанаваны ўласны варыянт алгарытма ПЗ (малюнак 2).



Малюнак 2. Алгарытм пераапрацоўкі паведамленняў

Алгарытм змяшчае наступную паслядоўнасць дзеянняў:

1. захаванне ўваходных паведамленняў у базе дадзеных;
2. спробуапрацоўкі захаваных паведамленняў па чарзе;
3. пазнакуня ўдалых паведамленняў з захаваннем кода памылкі і колькасці спробаў перавыканання;
4. паўторную спробу перавыкананняня ўдалых паведамленняў праз адтэрмінаваны час;
5. перавыкананне няўдалых паведамленняў, якія засталіся неапрацаванымі пасля кроку 4, з павялічанай колькасцю спробаў; пры перавышэнні гранічнадапушчальнай колькасці спробаў паведамленне пазначаецца як цалкамневыканальнае;
6. захаванне вынікаў апрацоўкі паведамленняў.

Прапанаваны аўтарам алгарытм паказаў магчымасць не толькі перавыконваць паведамленні, але і збіраць статыстыку па колькасці, тыпу, коду памылкі і частаце ўзнікненняня ўдалых паведамленняў.

Сабраная статыстыка здольна дапамагчы адшукаць недасканалыя месцы сістэмы праз паведамленні, якія пазначаюцца няўдалымі найбольш часта, а таксама праз мікрасервісы, якія іх дасылаюць. Праз аналіз названай статыстыкі магчыма выявіць архітэктурнацілагічна памылкова пабудаваныя часткі коду, што маюць неабходнаць у пераглядзе і ўдасканаленні.

Такім чынам, прыведзены алгарытм вызначыў наступныя якасці:

– магчымасць перавыкананняня ўдалых паведамленняў у папулярным брокеры RabbitMQ;
– збор статыстыкі па няўдалых паведамленнях у сістэме;
– магчымасць вышуку архітэктурнаці лагічна памылкова пабудаваныхчастак коду;
– падтрыманне базы дадзеных асобнаўзятага мікрасервіса ў актуальным стане нават падчас форс-мажорных абставін вакол іншых мікрасервісаў – адключэнне электрычнасці, парушэнне сувязі і інш.;
– магчымасць гібкай наладкі часу адтэрміноўкі і колькасці спробаў перавыканання спажывецкага паведамлення, а таксама дадатковых дзеянняў напярэданні і пасля яго апрацоўкі.

Літаратура

1. Микросервисная архитектура [Электроннырэсурс] / Вікіпедыя. –Рэжым доступа: https://ru.wikipedia.org/wiki/Микросервисная_архитектура. – Дата доступа: 31.03.2019.

2. Implementing an event bus with RabbitMQ for the development or test environment [Electronic resource] / February, the 10th, 2018. – Mode of access: https://docs.microsoft.com/en-us/dotnet/standard/microservices-architecture/ multi-container-microservice-net-applications/rabbitmq-event-bus-development-test-environment. – Date of access: 31.03.2019.

3. RabbitMQ для профессионалов [Электроннырэсурс] / 10 лютага 2018 года. – Рэжым доступа: http://onreader.mdl.ru/RabbitMQInDepth/ content/Ch01.html. – Дата доступа: 08.04.2019.