УДК 004.2

# MIGRATION OF HIGHLY CUSTOMIZED DESKTOP APPLICATIONS INTO CLOUD SERVICES

Lebed Y.N., engineer
CCSI Limited
Republic of Ireland, Dublin; Spain, Barcelona
yuriy.lebed@ccsi.ie

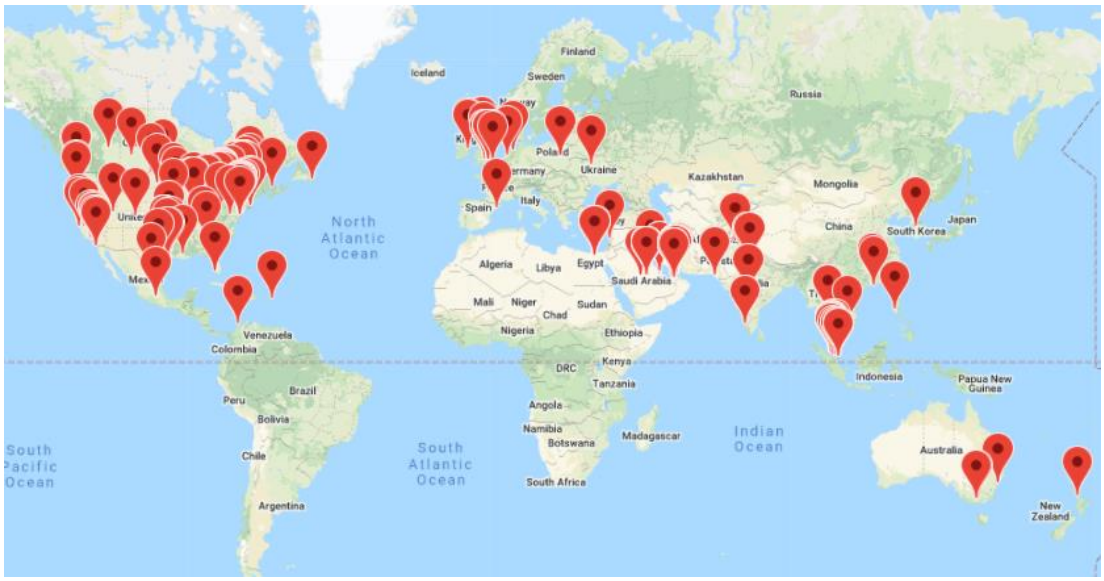
***Abstract.*** *During the development were implemented a few different solution architectures with analysis by the following criteria: final reliability, performance, expandability and customisation. The best results were achieved by architecture with microservices with generation of visual content.*
***Keywords:*** *solution architecture; software architecture; customisation; microservice.*

As the PC era started multiple aspects of businesses were attempted to be automated. The spreadsheets application really made a big push for areas where calculations were straightforward and not too complex. Next step, creation of desktop applications supposed to increase usability and durability. Over time some of them were picked up by big players and we can see accounting, warehousing, CRM, etc. standard applications broadly offered and used by millions of customers which contribute to their profitability.

On the other end, there were a number of applications which evolved into quite a knowledge or science-intensive software product with relatively narrow demand. Where institutional investments have no interest because of the relatively small client base and low cash flow.

But for those whose business niche was continuing to expand a desktop application becomes more and more difficult to maintain and enhance. The obvious solution is to go online [1]. Which solution architecture to choose?

What do we have to deal with? The research is based on software for the selection of air handling units and their components, multilanguage, multicurrency, 24/7 application, with users spread around the globe. Picture 1.




Picture 1. Users' locations.

Apart from this, the application uses circa forty third-party libraries some of them developed more when twenty years ago and have never been updated by the manufacturer. Some of them thread-unsafe.

The common solutions were:

- to convert the desktop app into a thick client with sufficiently reduces overheads for license management;
- thread-unsafe modules were wrapped by messaging queue clients which have eliminated interference between instances processed in parallel;
- the desktop's functionalities were one by one gradually moved to web-based services.
- the web core fully maintains user management and project management where the individual product functionalities were implemented using the following solution architectures;
- the software architecture chosen is MVC;
- where controllers implemented design patterns such as abstract factory, strategy, decorator, interpreter, command [2];
- coding practice follows the SOLID principles [3];
- types of databases used: SQL, no-SQL, blob storage.

The solution architectures were considered.

Standalone web application. An application where all functionality presents in one deployment.

Pros: easy deployment, requiring lesser developers' skills.

Cons: because the app is massive, practically not possible to scale dynamically, high probability for errors affecting all users, each change request requires full redeployment, adding new modules makes it even more massive, difficult to accommodate all third-party libraries inside one environment.

Standalone web application with scalable microservices. An application where performance-critical functionality is moved to easily scalable separate services.

Pros: split deployment, dynamic scalability, the lesser probability of fatal errors, individual containers for each third-party requiring a specific environment.

Cons: difficult to maintain seamless UI customization, still a lot of work to add new functionality.

Web portal with UI microservices. An application where all specific functionality and related UI content are generated on separate dedicated services.

Pros: all from above, plus comfortable and secure customization of UI and functionality.

Cons: more efforts to manage a larger number of software components, require a unified solution for localization.

***Conclusions***.

Development of the portal with UI microservices responsible for specific functionality allows to deploy independent parts of the application with minimum risk of fatal errors, makes easy deep customization including UI, and utilises scalability to keep performance at the required level.

## REFERENCES

[1] Small, M.J. (2018). *Stress Free Cloud Migration*. RMS Technology Consulting, LLC.

[2] Gamma, E., Helm, R., Jonson, R., Vlissides, J. (1994). *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley Professional.

[3] Martin, R. (2002). *Agile Software Development, Principles, Patterns, and Practices*. Pearson.

Yuriy N. Lebed

**Migration of highly customized desktop application into cloud services**

***Анотація.*** *У ході роботи було реалізовано декілька архитектур програмного комплексу та зроблено порівняний аналіз за критеріями: загальна остаточна надійність, продуктивність, придатність до розширення функціональности та налаштування для окремого користувача. Найбільш перспективною виявилися архітектура з використанням мікросервісів з генерацією візуального контента.*

***Ключові слова:*** *архітектрура рішення; архітектура програмноо забеспечення; кастомізація; мікросервіс.*