The use of microservices to implement cross process integration and data sharing

Balint Maschio
IT department Monrif S.p.A
balint.maschio@monrif.net

Abstract

This paper looks into the use of microservices from an industrial prospective; looking at the idea behind the use of microservices as software integration pattern in Monrif S.p.A.

1 Introduction

Monrif S.p,A is a company that operates mainly in the newspapers industry. This industry is facing a period of complex and challenging changes. The IT departments are playing an essential role in matching this challenge; yet the same IT departments find themselves to operate with smaller budgets. This new financial constraint has generated a shift in their project model: from big projects with many actors, to more lightweight, fast evolving projects.

In this paper we describe our experience in the adoption of the microservices approach to deal with this challenge, we consider this approach to be a potential candidate for creating a better correlation between how we think projects and how we actually implement them.

2 Software ecosystem

Monrif S.p.A. is a company with a very heterogeneous software ecosystem where many different applications and technologies coexisting. This kind of environment poses a challenge when cross processes interaction and data sharing becomes a must and application to application connection are not anymore a feasible answer. Often these connections have evolved from ETL techniques such as batch jobs and asynchronous procedures with replicated data structures making the integration process difficult to monitor and to make evolve. At this point, the IT department was faced with a choice, either continue in this way and add more and more complexity or switch paradigm and re-think how processes integration was done in our organisation.

3 Change in paradigm

Paradigm changes are often perceived, and rightly so, as expensive both in economical and organisation terms. In an application and processes integration project, this can be particularly true; adopting a SOA (Service Oriented Architecture) approach coupled with an ESB (Enterprise Service Bus) can lead to an increase in project complexity and to an increase in hardware and software infrastructure requirements.

In the autumn of 2015 the IT department of Monrif was asked to evaluate the opportunity of using an ESB with "off the shelf" solution such Talend or alternatively to implement a homemade microservices solution.

Although Talend is an excellent product with a proven history of successful projects, it was thought to be an intricate product due to the technologies and frameworks involved, which would have required a steep learning curve and intensive use of external consultancy.

The evident lack of technical expertise coupled and not sufficient level maturity of the IT department in handling the some of the design and engineering aspect of a SOA project made us not confident to start a solution development that would have required the level of project specification and project management commons in a SOA/ERP project

On the other hand, a homemade microservices solution was understood to be more manageable on the following lines of thinking:

1. We could approach the project dividing it into process bounded steps within a larger domain (ex : Purchase Chain , Plant Management)
2. Smaller sized project steps can be handled by a time constrained small team
3. The delivery of smaller incremental steps will facilitate the migration of the existing "application to application integration" to a more process integration environment by allowing the team to gradually switch off the old implementation and migrating to the new implementation
4. A smaller project can fail without considerable organizational or financial implications

The selection of Jolie as the microservice language was driven by the following consideration

1. A successful integration project already developed in Jolie present in the organization. This project involved the integration of SAPERP with DMS (Document Management System) using microservices that collected transformed and routed data between the two applications.
2. The small resources footprint required to run Jolie's microservices
3. The availability of lightweight enterprise microservices monitoring and control system.
4. Possibility of developing own languages libraries in Java.

4 Monrif's vision on microservices

The literature is full of excellent examples and definition on: what it is a microservice, what should it do and how to implement it; yet we believe that each organization should create its own "Vision" on microservice. Being domain driven design closely linked to microservices design it seems reasonable to think that the microservice vision Monrif may differ from that of companies like Uber or Netflix being different the domain scope and application requirements.

Monrif's microservice vision is based on this four families of microservices

1. Data service: those microservices that access application data via DB connection or application specific connector
2. Orchestration Services: those microservices that implement the business logic orchestrating several Data services to realise the desired process or functionality

3. Event Handler: those microservices that trigger certain behaviour as response of a specific request generated by a third party software using normally orchestration services and handling process level error
4. Scheduler service: those service that trigger certain behaviour at specific time of the day (asynchronous processes)

The data service constraints are quite stringent and we avoid to insert any process logic inside the operations implementation to maximise the data/process decoupling and trying to restrict the operation to the CRUD operation for the desired data structures

In the case of orchestrations or event handler family a single microservice may fall in both family because the event logic may be insert directly in the event handler operation. The rule that we follow to decide when separate the business logic as separated orchestration service is based on:

1. Does the business logic need to be used by more than a service consumer
2. Does the business logic frequently change
3. Can we group together similar operation based on their application domain

The Scheduler microservices are used to handle those asynchronous processes that can be run in the background. In Monrif we use this family of microservices to handle cross application data transfer.

5 Conclusion

After just about two year of microservices development our thirty-two active microservices, can be divided into the following domains of activity (Table.1) where we can also see an estimated number of messages exchanged between the microservices for a specific application domain.

Domain	Nr of active microservice	Nr messages monthly
DMS processes	10	˜350 K
Plant Management	7	˜90 K
Editorial activities	3	˜ 70 K
HR process	8	˜ 150K
Ecommerce	4	˜ 90K

Table 1 Number of messages for microservices.

As an overall result, considering factors such reliability of the services in terms continuous of up-time (avg. 5 month), number of exchanged messages per month, and the level of integration between applications , although the latter one more difficult to quantify, we consider ourselves satisfied with the choice of adopting the microservice paradigm.

Furthermore the adoption of the microservice paradigm has given the confidence to a small team, composed by four IT professionals with different backgrounds, to develop a workable integration strategy

The view on Jolie is overall positive thanks to its simple syntax and simplified message handling; although some residual perplexity exist on the level of maturity of the language, mainly connected to the absence of a “proper” IDE and other programming supporting tools.

Our personal experience seems to suggest that microservices development with Jolie can be undertaken by smaller industrial reality such Monrif, that previously may have been “scare off” by the daunting task software and processes integration.