Microservices-based **Architecture Deployment** for an International ICT Consulting Firm

ATTENTION. ALWAYS.

aspire SYSTEMS
attention. always.

THE CUSTOMER

Our client is a leading ICT solutions supplier in the mortgage and consumptive credit industry. Their job is to develop business-critical, self-supporting products that can also be embedded in other product suites. The company also offers enterprise-level agile coaching and development services.

THE CHALLENGE

The model before our solution implementation had independent features like transformation (service that converts business rules from one form to another), validation and building & extraction. While the features were separate, they were essentially deployed as a single system.

OLD SYSTEMS

SERVICE - 1	SERVICE - 2	SERVICE - 3 ..N
User Language	Feature	Feature
Transformation	Feature	Feature
Algorithm	Feature	Feature

BUSINESS CHALLENGES

The transformation component in discussion is a program to transform business rules from user's language to machine understandable pieces of code. As the component's usage increased, it required frequent pit-stops to update and this in turn made this whole product bulky and operationally cumbersome affecting the overall efficiency.

Another challenge in deploying these independent components inside the system is the threat to the operation of the entire program in case of any defects in its individual elements. This incurs process overheads for the teams that support and maintain such systems, especially as the organization grows.

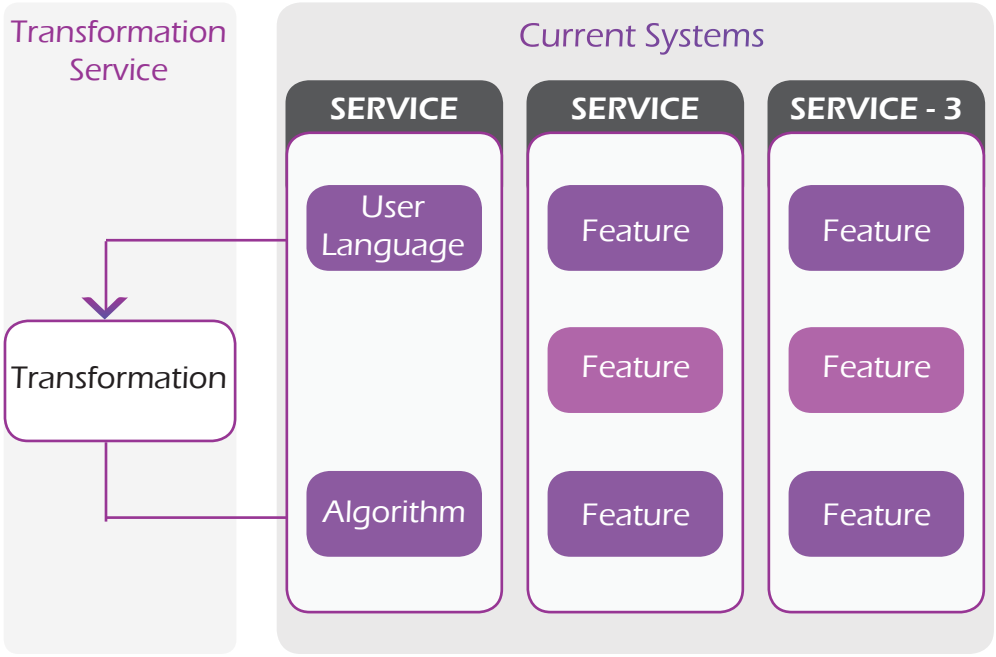
TECHNICAL CHALLENGES

This transformation component was initially thought as a "nice-to-have" solution. In reality, it proved to be more than efficient because of the flexibility it offered in letting the user write in the language of their choosing. Once this program scaled and the module was run frequently, it required frequent updates to perform in different versions. This created an operational overhead as the original design of that module wasn't scalable. With increased frequency of changes per version, this feature demanded a detailed release planning and contributed to the increase in bulk of the overall program.

THE SOLUTION

"Our solution essentially mitigated the load on the program, in terms of time, cost and efficiency, by letting the program micro-manage modules as microservices. This model not just increased the program's ability to multi-task, but also allowed easier defect detection and tracking."

The solution we offered to the client involved a revamp in the architecture, around the transformation component. We proposed to pull out the transformation service, along with a few other independent programs, and run it outside the system as a microservice.



TECHNOLOGY SNAPSHOT

Platform	Java
Core Framework	Spring
ORM	HIbernate
UI	GWT
Version Control	Git
Build Server	Jenkins
Repository	Nexus
Virtualization	Vmware
Issue Tracking	Jira

TARGETED POINTS

- Increased performance
- Achievement of stateless-ness
- Scalability
- Standardization of transformation interfaces
- Reusability

BEST PRACTICES

LOOSE COUPLING

HANDLING FAILURES

CUSTOMER BENEFITS

Identifying transformation component as a microservice helped the program to shed the process redundancies, improve the efficiency and cut the costs spent in additional maintenance and support of the system. Scalability became a breeze with this architecture as the entire module can be isolated and individual components can be upgraded without disturbing the environment of the system.

This solution also offered to the customers an increase in focus on the transformation component. Since it has been pulled out to run as a separate microservice, it came under the purview of an independent team which handled its specification and release. The cost incurred in maintaining an independent component was much less than the benefits that were derived in the process.

COMPARISON BETWEEN THE BEFORE & AFTER SITUATION

- The critical processes of a program like planning, change and optimization, which were previously tough to handle, are now managed efficiently because of the transformation of the components as a microservice.
- The solution changed the status of the transformation component from “technical service” to “small product” which resulted in increased attention to it.

HOW DID THE SOLUTION WORK?

The solution identifies stronger module boundaries through the transformation service as a microservice with independent deployability and lets the product call it whenever it needs it as opposed to integrating it with the entire system.

FUTURE IMPACT

The isolation of this component has opened up a wide room for scalability in the future. Its independence also lets the customers to accommodate the growing system of microservices as their business expands.

www.aspiresys.com

aspire SYSTEMS
attention. always.

ATTENTION. ALWAYS.

ABOUT ASPIRE

Aspire System is a global technology services firm serving as a trusted technology partner for our customers. We work with some of the world's most innovative enterprises and independent software vendors, helping them leverage technology and outsourcing in our specific areas of expertise. Our core philosophy of "Attention. Always." communicates our belief in lavishing care and attention on our customer and employees.

NORTH AMERICA | UK | SINGAPORE | BENELUX | NORDIC | MIDDLE EAST | INDIA
+91 - 044 - 67404000, +1 - 630 - 368 - 0970, +44 - 203 170 6115, +65 31633050

For more info contact
info@aspiresys.com or visit www.aspiresys.com