Digitizing contract management with microservices on Azure

At Microsoft, we've created a microservices architecture and micro front-end architecture that uses Microsoft Azure to provide an improved experience with our business contract life cycle management platform. The new solution provides a simplified interface for our users and hides the complexities of the underlying contract-management platform. The new solution digitizes our contract-management processes, which greatly increases process efficiency. It also takes advantage of Azure components to create increased reusability and scalability so that our solution can change and grow with our business.

Understanding contract management at Microsoft

Contracts are at the core of Microsoft business. Each year, we negotiate service agreements with a large array of vendors to provide products and services that help us run our business. Behind every agreement—these define the terms and create the framework within which we receive and pay for these products and services—is a contract. Microsoft manages more than 1.1 million contracts. Contracts contain commercial terms, payment terms, pricing, renewal information, obligations, incentives, risk details, liabilities, and other attributes. This contract information is used by the Microsoft legal, procurement, sales, and other divisions to facilitate more intelligent decision making.

We use a third-party service to provide contract cycle management services, which cover the breadth of contract-management tasks at Microsoft. Our contract-management services operate on a massive scale. Here are some key metrics from our contract-management environment:

- Approximately 1,000 unique users each week
- 100,000 contracts created each year
- More than 20,000 outside views per month
- Approximately 400 attributes per contract

Identifying room for improvement

Historically, our contract life cycle management provider supplied a complete solution for Microsoft contract-management needs. Typically, these solutions didn’t enable digitization of the contracts and they didn’t provide an experience tailored to our business needs. This resulted in redundancy and inefficiency in the system. A lack of end-to-end digitization hindered process automation and improvements in efficiency, compliance, and risk management. We recognized that a much of the information in contracts wasn’t being tracked, such as security and privacy terms, payment terms, pricing, renewal information, obligations, risks, and liabilities. All this data could enable various parts of the organization to ensure more efficient contract management and intelligent, data-driven decision making.

We reassess our needs on an ongoing basis, choosing the provider and solution that best fulfills our business requirements. With each solution switchover, our contract-management team facilitates the technical and functional migration to the new contract lifecycle management platform. Often, this process is time consuming for our technical teams and awkward for our contract-management employees who use the contract-management solution. A few of the key issues include:

- We didn’t have a unified method for digitizing the contracts. Each step in the digitization effort was time consuming, inefficient, and had potential for human error.
- We had no insight into the contract-management data. Our processes relied on manual reviews and audits, which barely covered 1 percent of our contract portfolio.
• It was impossible to determine the potential risk or leverage to which Microsoft was exposed based on contract contents. Obtaining that information relied on Manual Reviews and Audits, which couldn’t handle more than 1 percent of Microsoft contracts.
• Our users needed to learn a new tool and adapt to a new user interface each time the contract-management solution changed.
• Our users and customers were exposed to the contract-management solution’s complexities, which hindered their focus on what was important to them.
• We needed a greater source of knowledge to enable our users to create more insightful and manageable contracts.

Soon after our most recent solution provider was selected, we recognized the need for a better end-to-end experience with intelligent, automated contract management. We were preparing for another complex migration, and our users faced the challenges of learning yet another user interface and set of processes. We realized that we needed to create a new scenario in which our users and support staff were removed from the complexities inherent in our previous solution-adoption processes.

Creating intelligent, automated contract management

To resolve existing issues and improve the contract-management experience across the organization, we examined the primary problems within the contract-management environment. We then created an approach that addressed each of the major issues and provided potential solutions. Our overarching goals were to simplify and digitize the contract-management process and build intelligence into the solution to serve the Microsoft organization’s compliance and risk-management needs. Within this framework, we established more specific goals, including:

• Create a seamless, self-service user experience across any contract-management solution.
• Simplify and standardize maintenance and processing tasks.
• Develop a solution with maximum reusability and modularity.

Using a modern contract-management solution

Onboarding to a modern, API-based contract management solution with capabilities like template automation, obligation management, and workflow management was a key objective in our solution. We’ve adopted the Icertis Contract Management (ICM) platform, which is natively and exclusively built on Azure as a platform as a service (PaaS) application, for our contract life cycle management needs. The ICM platform’s best-in-class flexibility, usability, and open API structure is the ideal technology to migrate more than 1 million Microsoft contracts and help build a truly unified contract-management experience as part of the Microsoft digital transformation journey.

Creating a reusable user experience

Our primary design strategy was to create for our users a streamlined experience that we could use to interact with a headless contract-management solution. One of our biggest issues was the lack of a coherent, familiar user experience. By creating a reusable, independent UI, we were able to give our users a consistent experience that interfaced with the cloud-based contract-management solution we used.

Building modular functionality with microservices

We created the core functionality of the solution with a set of REST API-based microservices hosted in Microsoft Azure that interfaced with our new API-based contract management solution. We created these APIs based on our business needs and the core design of the existing solution. We used the APIs to create a layer into which we could
attach any type of functionality that we wanted to build on top of the third-party solution, such as user interaction, security, search, or data retrieval.

We built our APIs to be modular and we integrated them by using a microservices architecture. Each interaction function of interaction was created as a standalone REST API, with integration points for other microservices. These microservices act as the integration point not only for the user interface but also for any other external systems that must integrate with our contract-management solution. The core microservices included:

- **Contracts.** The Contracts API provides core contract-management functionality, including contract creation, updates, and workflow tasks.
- **Case Management.** The Case Management API enables users to raise a request to contract specialists to get help with contract creation, negotiation, or other legal services.
- **Search.** The Search API integrates directly with the contract-management solution’s search engine and exposes it to a variety of important search functions.

We’re leveraging native Azure tools, including AppInsights and Azure Monitor, to provide complete solution telemetry. We use that telemetry data to improve processes and identify issues within the architecture.

**Building user experience with micro front-end architecture**

We designed a series of UI objects coupled loosely to provide a core UI. With this method, we created individual micro front-end components that provide a set of contract-management interfaces that can be integrated to create a targeted user experience. We used the React framework to create several micro front-end components that provide the core user experiences. These components connect with the microservice APIs to create the end-to-end user experience.

We’ve used this model to create an independent interface that we can customize and modify to fulfill our business requirements. Each micro front-end component can be displayed in several different ways and combined with other micro front-end components to create a more comprehensive UI tailored to the user roles within our contract-management environment. Because of its relationship with the microservices, the UI remains consistent for our users. Our core micro front-end components include:

- **Contract creation.** Tools used for basic contract creation and modification tasks.
- **Contract action list.** A list of contracts that need action from users or contract facilitators.
- **Contract search.** A search interface used to locate contracts or contract information.
- **Notifications.** A list of current notifications providing information on important aspects of the new solution.

We’ve also integrated the front-end components to simplify more complex contract-management tasks. For example, when a user initiates a request to create a contract, the system prompts for necessary information. It then begins the creation process and sends a case-management request to the contract-management team, which starts the process with all the information that they need to continue creation. This enables users to remain in a single interface instead of hopping between interfaces, following up, and performing several other process steps that were necessary with the previous solution. We also built our interfaces with accessibility in mind, ensuring that we provided a complete experience for our employees, regardless of how they access the interface.

**Implementing modern engineering practices**

We adopted modern engineering practices to develop our solution, embracing a DevOps approach to engineering methods and life-cycle management. Our approach involves frequent, small iterations developed with the direct involvement of our business stakeholders to ensure rapid and agile development of a solution that solves relevant business problems.

**Building with security and instrumentation**

We integrated modern security engineering elements into the new solution, ensuring that security was built-in from the beginning of the development process. We adopted a modular approach to security, by using the core security
elements from the third-party contract-management platform. We also extended functionality with a security library that integrates with each microservice in the architecture.

Our hybrid security layer provided additional security capabilities, integrating with Azure Active Directory through Microsoft Graph for identity and authentication management and to leverage preexisting user metadata. The security layer enables the solution to assign distinct layers of access management based on role, group membership, and metadata attributes. We leveraged native Azure monitoring components, including Azure Monitor and Application Insights to create instrumentation and alerting for the entire microservices architecture.

Migrating between contract management platforms

We also built our solution to provide a better migration and integration experience between contract-management provider platforms. To assist with our most recent contract management provider migration, we've created an integration layer than enables us to send information between the previous and current platforms.

Part of our migration process involves preserving the state of the previous platform in case rollback or access to preexisting data is required. We implemented Azure Service Bus to facilitate messaging between the previous and current platforms, providing live updating of both platforms as changes to the contract-management environment occur. This integration layer provides absolute traceability for the data that has been migrated between contract management solutions, thereby easing the reconciliation process.

Figure 1. The microservices and micro front-end architecture

Benefitting from a reusable experience

Our new solution required only seven months to implement, from the beginning of development and design to complete rollout. We’ve achieved our primary goals and created a solution with several important benefits:

- A loosely coupled, flexible microservices architectural pattern.
- A flexible platform that is built to change and adapt.
- A fluid, responsive UI with minimal hops.
- Increased reusability for components.

microsoft.com/itshowcase
• Automated contract-management processes, including contract creation and legal review. The integration and experience that our solution provides also created tangible benefits for our contract-management teams, including:

• **Reduced contract-creation time.** With the new UI, the time it takes to create a contract has been reduced by approximately 50 percent.

• **Improved back-end processes.** With automation and shared data between microservices, our support staff has a more complete information set when resolving issues. We’ve reduced the time it takes to resolve issues by almost 25 percent.

**Moving forward**

Our goal is to build a single contract life-cycle management model by continually improving user experience, identifying crucial issues early in the contracting cycle, and, above all, reducing time by automating the contract-generation process for all contract types. We continue to create new integration and innovation within our microservices and micro-UI architecture. We’re currently in the process of testing the adoption of new services and interfaces, including:

• A contextual microservice that provides context-based information to the user to assist with or simplify tasks.

• A cognitive microservice to extract relevant data from contract-management processes and expose details, trends, and insights by using AI and machine learning.

• A configuration interface to enable configuration of the contract-management platform within the new UI, including rule management.

The modular nature of the new solution makes it much simpler for us to add functionality and integrate new elements of interface design. We’re examining possible improvements to overall solution intelligence with new AI and machine-learning functionality that will extend our contract-management platform’s capabilities. We’re also examining opportunities to leverage the solution elsewhere in the organization and extend the microservices and micro front-end architecture to enable other parts of our business to operate more efficiently.

The contract life-cycle management experience is the best that it’s ever been at Microsoft. With our new solution, we’ve created an agile, more fluid interface and process for our users. The underlying contract-management platform is easier to manage and migrate, and the migration process is transparent to our users. We’re saving time and money with improved processes, and we’ve created a solution that’s built to expand and grow as our contract-management needs change. We’re looking forward to improving the platform, extending its capabilities, and bringing it to new areas of business within Microsoft.

**For more information**

**Microsoft IT Showcase**

[Microsoft.com/itsshowcase](http://microsoft.com/itsshowcase)

Finance uses anomaly detection and automation to transform royalty statements processing

Using AI and chatbots to simplify finance tools at Microsoft

How Microsoft used SQL Azure and Azure Service Fabric to rebuild a key internal app

© 2019 Microsoft Corporation. This document is for informational purposes only. MICROSOFT MAKES NO WARRANTIES, EXPRESS OR IMPLIED, IN THIS SUMMARY. The names of actual companies and products mentioned herein may be the trademarks of their respective owners.