Redesigning business applications at Microsoft using PowerApps

Microsoft IT is using PowerApps to offer a new employee experience during common activities. Thrive, our company dashboard app, uses PowerApps to unify the corporate app experience with a consistent user interface across traditional lines of business. For the engineers building this experience, it provides easy-to-design app components and a single point of configuration and control. PowerApps makes it easy to incorporate our existing app portfolio within the Thrive experience and to build new productivity solutions for our employees.

Understanding mobile apps at Microsoft

At Microsoft, our employees can access apps that allow them to do hundreds of different tasks. From tracking and reporting vacation hours, to checking for meeting room availability, to formally acknowledging the outstanding effort of a colleague, our app ecosystem is broad and diverse.

Over the past few years, our cloud-first, mobile-first strategy has been the catalyst for an explosion of different web and mobile apps that enable our employees to work wherever they are. As this app ecosystem has grown, so have the number of developers creating apps and business groups that are championing app usage. The result has been poor discoverability (particularly from a mobile device) and inconsistent user experiences, which has contributed to low adoption of these apps.

We've recently made plans to offer a better web and mobile app experience for our employees and we've set some important goals to increase employee engagement and productivity:

- Put users at the center of the experience and erase traditional boundaries around how we use apps.
- Provide a consistent user interface that integrates data and navigation across apps.
- Use existing technology to accelerate the development progress.
- Delight users with apps that anticipate their needs and help them reach their goals.

Choosing PowerApps

We selected PowerApps as the target platform for our app unification project. PowerApps gave us the flexibility to merge our apps into a single experience, offer a user-friendly and attractive UI, and integrate with other common cloud and mobility data sources.

PowerApps is a collection of software services that offer a more direct, simple way to develop and integrate apps. It connects to existing cloud services and data sources and helps to quickly build apps to meet specific needs. PowerApps also offers enterprise-grade capabilities combined with easy-to-use authoring. PowerApps consists of four major components:

- **web.powerapps.com**. This is the portal where you manage and share the apps you build.
- **PowerApps Studio**. You use PowerApps Studio to build powerful apps with easy-to-use visual tools.
- **PowerApps Mobile**. Apps developed in PowerApps run on mobile devices using PowerApps Mobile. Use PowerApps Mobile to run apps on Windows, iOS, and Android devices.
- **PowerApps admin center**. Use to administer PowerApps environments and other components.

Enterprise design process

Our design process centers on our users. We look at metrics from our internal portals and Helpdesk tickets to determine what users are searching for and the problems they encounter with our existing apps.

Once we identify an app that is a candidate for the mobile experience, we:

1. Develop a high-level scenario description using PowerPoint. This describes a persona (such as Susan, the manager) and what she is trying to do (for example, reviewing and approving vacation requests).
2. Create a mid-level composite of the app with Balsamiq, a third-party tool. We create mockups of the app screens, controls, and interactions, and diagram the workflow between screens.
3. Build a low-level composite with Marvel, another third-party tool. This near pixel-perfect representation of the app can be viewed on a mobile device and lets our team see what the app will look like before we build it.
4. Test features with the user community. Microsoft has a community of volunteers, Microsoft Elite, which review design concepts and vote on their favorite approach.
5. Review and iterate. Our team evaluates the approach and gives feedback. This team includes API developers and user interface (UI) developers.
6. Define the APIs necessary to support the solution. Our priority is reusability—APIs should be as generic and atomic as possible so that they can be reused for other solutions.

Implementing our design

When it's time to implement our design, we have a user experience (UX) designer create our image collateral. The designer creates SVG or PNG images and attaches them to the work item in Visual Studio Team Services (VSTS), along with links to our Marvel dynamic composites.

An API team develops Azure web APIs using Visual Studio. They write code in C# and we create a swagger UI for each endpoint so that we can test and troubleshoot our solution. We've configured VSTS so that our dev environment has continuous integration builds with Azure Resource Manager templates and our user acceptance testing and production environments are updated with explicit approval. PowerApps connection objects are created by the API developer and shared with the app maker.

We then have a UI designer build the app to match the dynamic composite and implement the connection objects. One of our teams has the UX designer create the shell of the app in the PowerApps design studio. The studio is simple to use and the designer can drag and drop UI elements and wire up the navigation. Then the designer turns it over to a UI developer to connect the data sources and add application logic.

Performance is important, so we have sub-second targets for our APIs. Each app has a loader screen and we fire our APIs in parallel by calling them from individual timer controls. For a more native-like experience we use the LoadData and SaveData functions to save non-sensitive data locally.

Thrive architecture

The Thrive architecture is an integrated collection of apps that come together in PowerApps and appear on the Thrive user interface.

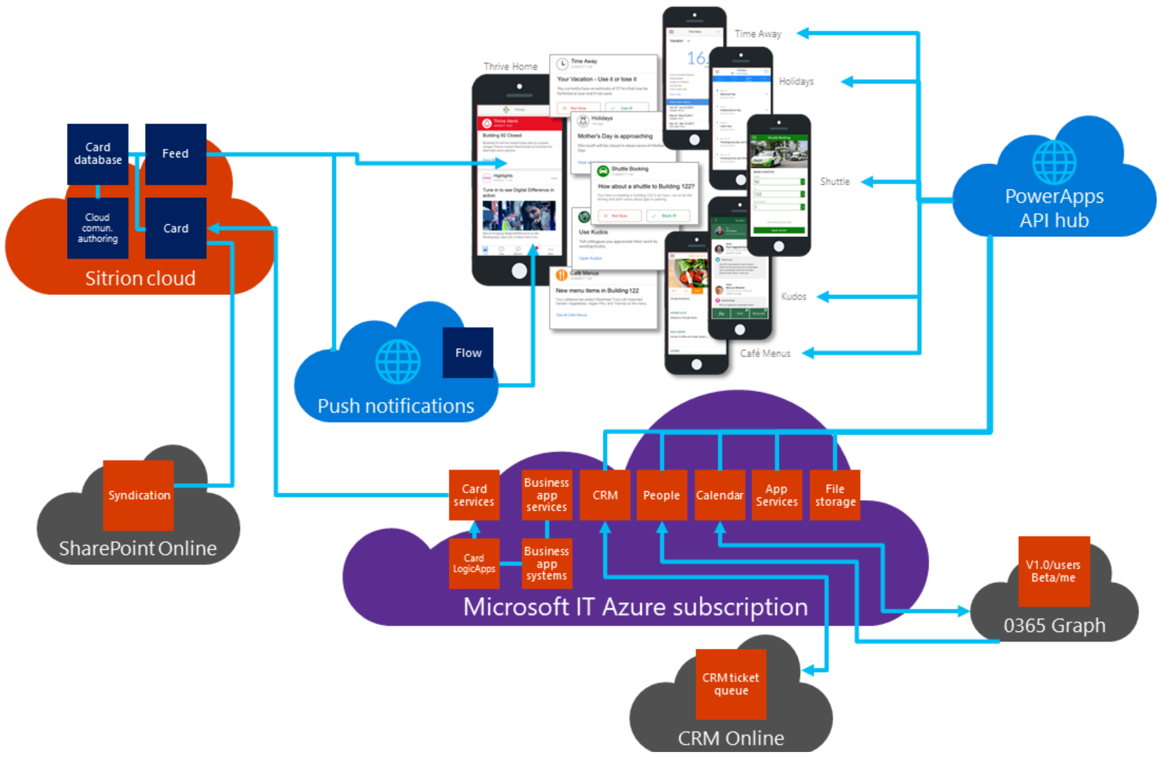


Figure 1. The Thrive architecture in PowerApps

The following components make up the functional Thrive ecosystem:

- PowerApps platform. Hosted on Azure, the PowerApps platform gives the app designer the plumbing, infrastructure, and services to create and support mobile apps.
- PowerApps player. Available for Windows Phone, Android, iOS, and desktop browsers, the player authenticates the user and serves up the application experience.
- Thrive Home app. The Thrive Home app has been named the Hero app for our tenant, so it's the first application our users see when they open PowerApps. This app has a feed that is served by Sitrion. Logic apps running in Azure monitor events, such as a payroll run, and create a card for each employee. For example, notify them that their paycheck has been deposited. Cards can simply serve up content and links, or they can be task cards. We have a separate tab in the app for task cards.
- Internal business apps. We have several apps specific to a business area or function. For example, we have apps to submit time away, for global holidays, kudos, employee feedback, paystub, and others. While they could be a single app, we kept them separate so that we could have different business teams work on them independently.
- Sitrion. This third party offers a service that creates cards in the productivity stream and targets them to users or Azure Active Directory (Azure AD) security groups. Advanced capabilities like Channels, where we can push cards, allow us to further refine and personalize the stream.
- Thrive Platform APIs. We have built a collection of common APIs that can be used by any PowerApp in Thrive:
  - People. This API has common operations for people at Microsoft. We can get the user's profile, their org hierarchy, frequent contacts through Office Graph, cached photos from the profile store, and other data.
  - Calendar. Common calendaring functions that talk to Office Graph. We can read and write to a user's calendar, set their out of office automatic reply, and other tasks.
  - Artifact. We use this API to upload binary files to Azure blob storage and secure them to an access control list. This stores audio files for our Kudos app and screenshots for our fitness reimbursement program.
  - AskHR. This API lets us create CRM Online tickets directly from our PowerApps.

    - AppServices. We use this API to count all of the Thrive PowerApps per environment. This is a list of apps, app icons and descriptions, the link to the app, the security groups that it belongs to, and its current state.
    - AppSettings. Every app can call into this API to get app-level or Thrive-level settings. For example, we'll pull the current version number and release notes for an app, or things like email body text.
    - UserSettings. Some apps need persistent user settings, like whether a user has accepted the most recent version of the privacy notice or has seen the What's New screen of an app on first launch.

- Internal business APIs. Teams that want to add a PowerApp to Thrive will need to expose an internal business API. This allows operations to read and write from their systems. For example, we have an API that tunnels back into our corporate network via ExpressRoute to communicate with our time entry system.
- Flow. Sitrion created a webhook that calls the Thrive push notification flow. The payload has a card ID, target people and security groups, and notification text. The flow sends a notification to a user's mobile device with a link to Thrive Home and the card ID as a querystring parameter. The Thrive Home app scrolls to the target card upon launch.
- Azure Logic Apps. We use several Logic Apps to connect to data sources and generate cards. For example, we query our Time Away system for a list of users that will lose vacation at the end of the year if they don't use it. Each user receives a "Use it or lose it" card encouraging them to take their PTO, along with a link to our Time Away app.
- SharePoint Online. We use Thrive for content syndication from SharePoint Online. A collector, offered by Sitrion, monitors SharePoint lists for changes and publishes the content to a channel in our productivity stream.

User experience

We deliver Thrive as a collection of PowerApps, with Thrive Home as a hero app for easier discoverability. Using Logic Apps and event-based processes, we call the Sitrion API to create cards on the user's productivity stream. Using the Sitrion app builder, we create card templates for our various groups and solutions.

- *Information* cards display content to the user and may link to web pages or other apps.
- *Alert* cards are content cards that remain at the top of the stream, which we use for important notifications, like a building closure or important event at Microsoft.
- *Task* cards appear in the stream but also show up on a tasks tab and are only dismissed when the task is done.

We can push notifications to a client by specifying the right attributes in the JSON (JavaScript) payload when we create a card. This works with all three card types. A Sitrion webhook calls a flow within the Microsoft tenant to push a notification to a list of email addresses or security groups. The notification is configured to launch Thrive Home and pass in the card ID as a launch parameter so the user will go right to the target card.

Thrive Home has a loader screen where it calls our platform APIs and the Sitrion API. We do this in parallel with timers to optimize performance. A call to the Azure App Service API determines whether the app is online and won't call any more APIs; if not, it redirects to a maintenance screen. A quick call to App Settings and User Settings determines the current app version and if the user should be directed to a tutorial screen.

The Thrive Home app also has an Apps tab, populated by the AppServices API. It counts the other apps that the user has access to, and offers a quick link to them. Our internal business PowerApps act similarly on launch, except that they will call specific business APIs in addition to the platform APIs. Depending upon the action taken within an app, the back-end APIs may revoke a task card if completed or generate new cards based upon transactional events.

Security and privacy

Each layer of the solution was built with security in mind. Upon installation from the mobile app store, the PowerApps player asks the user for their work email address. This redirects the user to the Azure AD log-in screen for that enterprise. If multi-factor authentication is enabled, the user will be prompted for secondary authentication, like touch ID via the Microsoft authenticator app.

Upon successful authentication, the user is prompted to allow push notifications to the mobile device. Note that this authorization applies to all PowerApps within that enterprise, not app by app.

When an app, such as Thrive Home, is first downloaded, the mobile player will create connection objects on behalf of that user. In our case, we've configured single sign-on (SSO), so the Azure AD bearer token will be passed from the mobile client through the PowerApps API Hub to our API tier. If SSO is not configured, then the user will be prompted with a disambiguation dialog box so they can sign into the connections individually. If the application requires access to mobile capabilities such as the microphone or location services, the user will need to consent to those services.

Thrive applications are secured by granting View permissions to security groups. Within Microsoft, we have security groups that cover geographies, divisions, and roles, so we can scope our applications to the right audience. Users outside of that audience won't even see the applications. Similarly, we add these same security groups to our App Services so that a user can see only the Thrive apps that they're supposed to see and can quickly launch those apps. This service secures access to different components of the app, but it doesn't provide comprehensive security for the app itself.

Thrive applications, in turn, call our custom APIs. These APIs are secured by OAuth 2.0, receiving the Azure AD bearer token from the PowerApps API Hub. Since the APIs need to be publicly accessible we ensure that all security measures are in place at this tier. Some of the things we do include:

- The APIs reject any connection that does not authenticate with Azure AD.
- Operations are scoped so that a user can only interact with their own data.
- Error messages are sanitized so that they don't reveal details of the underlying systems.

Our APIs run under a specific security context when accessing data. Ideally, the app continues to use the user's permissions and accesses other systems using this minimally privileged account. Sometimes this isn't possible, so we'll use a trusted subsystem model and access systems with a restricted privilege domain account. Our apps display a privacy notice and, in some cases, we allow the users to explicitly opt-in to a data collection.

Integrating with third-party solutions

Thrive can integrate with third-party systems to allow highlighting within the application infrastructure. The key foundational discovery API is supplied by Sitrion, an infrastructure they market as Sitrion ONE. Since it's such an important element, we asked Sitrion to adapt it into a service we could use for our employees and share with other Microsoft clients using AppSource so they can build the same type of experience.

Given the nature of PowerApps and the Thrive architecture, other third-party systems can easily be integrated and used privately, publicly, or both. We expect to have more third-party app integrations that represent key value areas of our employee's experience. However, we're careful to balance app discovery with ease of use and simplicity in PowerApps. Figure 2 shows how apps are integrated in Thrive.

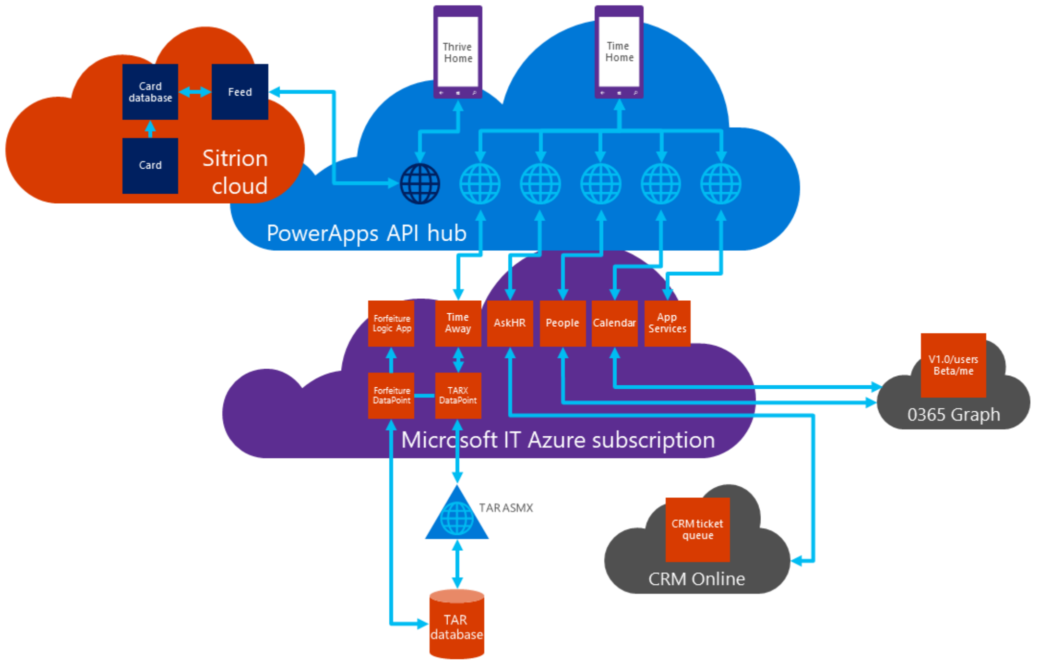
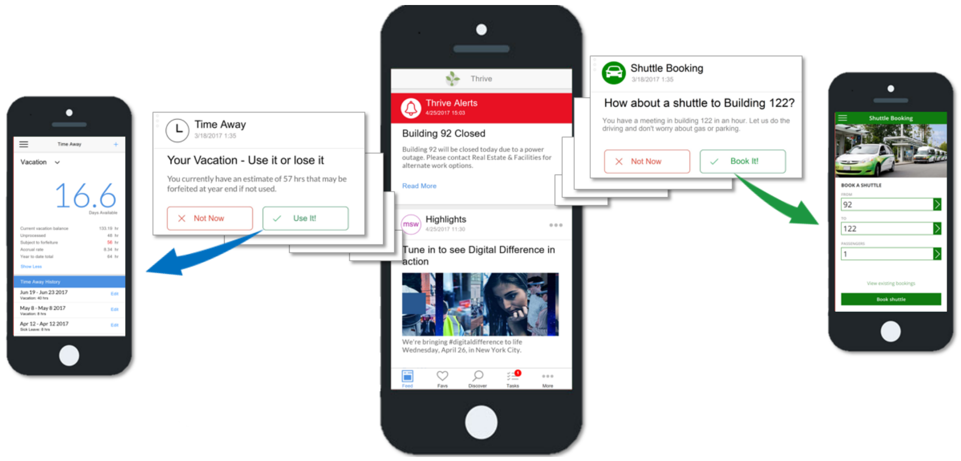


Figure 2. An example of app integration with Thrive (NOTE: TAR is time and absence reporting.)

Current state

Our employees use Thrive to read company news, get alerts about company events and priority items, or subscribe to content channels. Thrive can notify people about key transactions they may be unaware of, such as time off that they might be forfeiting, approving time away globally, viewing holidays around the world, and adding them to their calendar. Employees can also offer positive peer-to-peer kudos, manage their employment data, manage their profile, book time off and notify their team. Thrive is used for these tasks, along with several others and many more to come. Thrive Home is a central experience for employees that ties them into other productivity apps.

The Thrive experience is designed to be as seamless as possible for Microsoft employees. Although Thrive refers them to many apps, we try to limit the definition and identification of apps and app names that they see. Thrive provides the user interface and the app provides the underlying functionality. To the employee, it still appears that they are in Thrive; they haven't selected a different app or clicked on a different icon. They have simply searched for and found a task they want to perform and Thrive takes care of getting them the app functionality they need. Figure 3 shows different views of the Thrive interface on a mobile device.


Time Away
Vacation
16.6
Days Available
Time Away History
Jun 19 - Jun 23 2017
May 8 - May 8 2017
Apr 12 - Apr 12 2017

Time Away
3/18/2017 1:35
Your Vacation - Use it or lose it
You currently have an estimate of 57 hrs that may be forfeited at year end if not used.
Not Now
Use It!

Thrive
Thrive Alerts
4/25/2017 15:03
Building 92 Closed
Building 92 will be closed today due to a power outage. Please contact Real Estate & Facilities for alternate work options.
Read More
Highlights
4/25/2017 11:30
Tune in to see Digital Difference in action
We're bringing #digitaldifference to life Wednesday, April 26, in New York City.

Shuttle Booking
3/18/2017 1:35
How about a shuttle to Building 122?
You have a meeting in building 122 in an hour. Let us do the driving and don't worry about gas or parking.
Not Now
Book It!

Shuttle Booking
BOOK A SHUTTLE
FROM 92
TO 122
PASSENGERS 1
View existing bookings
Book shuttle

Figure 3. The Thrive app user experience

Future state

Thrive will continue to be the home for our employees to complete common tasks. We will also shift more of our focus to broadening the reach of Thrive and investing in technologies that help us predict and learn the needs of our users. Our productivity goal with Thrive is to create a threaded experience that helps users achieve their goal, not just individual tasks.

While we have focused on the mobile experience to begin with, we will bring this same unified approach to desktop experiences in the future and will enhance the overall experience with virtual assistants and bots where they are relevant. The key to Thrive will continue to be putting users in the center of the experience and using current and emerging technologies to make them more productive.

Benefits of using PowerApps

While Thrive is an ongoing project, we have already recognized several benefits of using PowerApps to create the Thrive experience for our employees:

- Rich, visually compelling UIs. The Thrive interface is easy to use and designed for both form and function. Our UX designers can create the app shell without any development experience.
- Rapid Enterprise Mobility development. We can create and deploy an app to our employees without having to go through a review and release exercise with mobile stores.
- App segmentation. We can make several apps appear as one, allowing individual teams to build and snap them into our ecosystem. The solution can grow without redeploying the entire experience each time.
- App sharing is easy. PowerApps integrates with Azure Active Directory to quickly share apps with security groups.
- We can redefine the patchwork quilt of legacy applications. If our legacy apps have a service tier, we can quickly build a series of PowerApps that have a common UX, creating a more seamless experience for employees.
- We can build it once for multiple platforms. We can create one app and use it on iOS, Windows Phone, Android, and the desktop browser.

For more information

Microsoft IT

http://www.Microsoft.com/ITShowcase