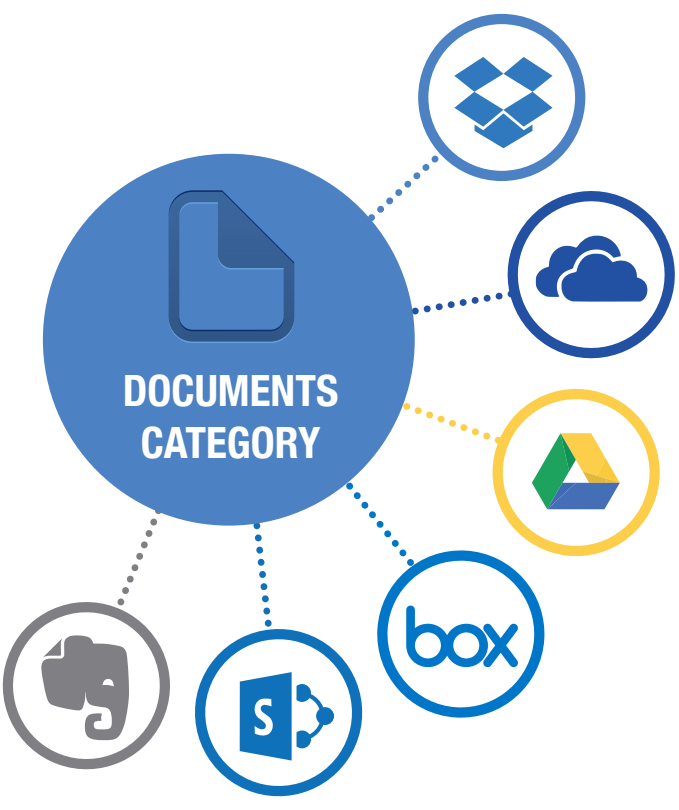
A PRODUCT MANAGER'S GUIDE TO CLOUD INTEGRATIONS

5 STEPS TO BUILDING COOPERATIVE APPS



STEP 1 Start With Categories Not Individual Endpoints

Your clients will always want options. Options are the "delighters" when it comes to building an optimal user experience. And knowing that your app exists within a vast ecosystem of cloud services and applications that your clients are already using, means there's vast opportunity to delight.

To start, identify the categories that your application needs to collaborate with such as documents storage, CRM, Finance or marketing automation systems. Do this instead of focusing on individual endpoints. When you think in terms of a category you consider use cases that broaden the options for your clients beyond designing to an individual end-point. For example, you will have clients who have document storage services (the Category) such as Dropbox, Google Drive, Box and more (the endpoints). Design your integration by considering the data objects and methods common to the category not just one service. A category approach will have your app cooperating more quickly with a wider range of services, thereby expanding your market share.

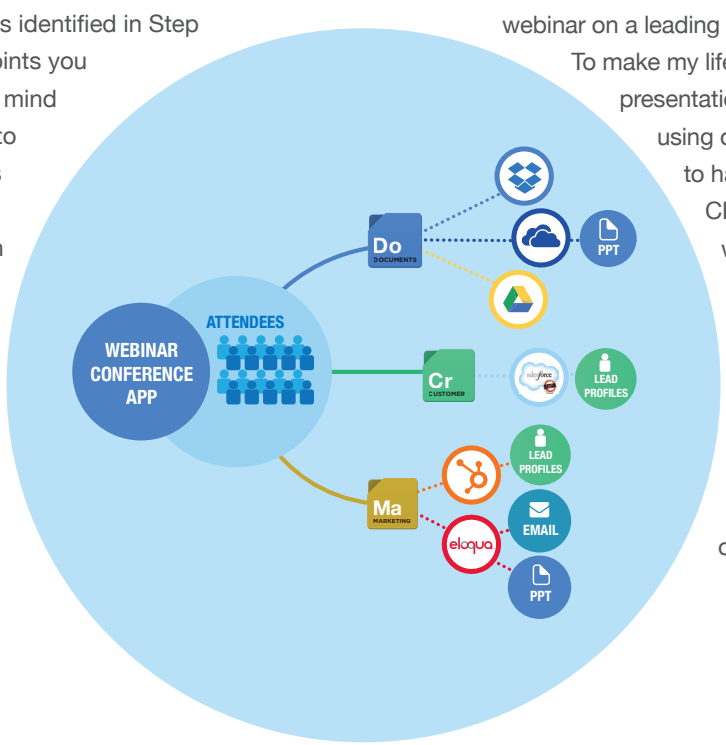
STEP 2 Collect Data to Prioritize Endpoints Within Each Category

Next, start by supporting the services that will give you the broadest market reach. At a leading job board in a company based in the UK, their users' data were primarily distributed among Google Drive, Dropbox and Microsoft OneDrive for cloud document storage. By collecting this data from the sales and business development team and then organizing into a single cloud document category, the priority individual endpoints became prevalent.

> "Avoid designing to one individual end-point and also avoid designing to all 100 end-points spread across a dozen categories."
>
> **Mark Geene,**
> CEO and Co-founder,
> Cloud Elements

STEP 3 Draft Your Cooperative App Model

An application doesn't exist on its own. Create a model that depicts the ecosystem of integrations at your organization. Organize it by the categories identified in Step 1 and the prioritized end-points you gathered in Step 2. Keep in mind that integration is a means to an end, and collaboration is your goal. The ideal user experience is an application that seamlessly interacts with the data your customer has stored in other applications while never leaving your app. Focus on the benefits your applications users will receive by cooperating with other apps in the category you selected.



Here's a use case to hit this topic home:
As a marketer, I want to promote, launch and execute a webinar on a leading web conferencing application. To make my life easier, I'd like to share the presentation via Box after the webinar using our marketing platform. I aim to have my email lists stored in our CRM, registrants tracked in the web conferencing app, and activity cards updated back in the CRM as they attend the webinar. Integrating these applications enables me, as a marketer, to effectively nurture new leads and convert to qualified opportunities for sales.

STEP 4 Develop Integration Use Cases

Integration use case models typically follow a consistent pattern. You can use this pattern to guide the definition of the user stories required to develop the cooperative use cases for your application. Discover your application's integration pattern by evaluating these 10 dimensions of integration use cases:

SELECT
Where will my app present the user interface to select the end-points that your user will connect with your app?

AUTHENTICATE
How will I manage the authentication for each end-point that I'm connecting with my app. What type of authentication mechanism does the end-point use (e.g., OAuth, SAML)? Where will I store the keys and refresh keys?

OBJECTS
Which standard objects does my application need from each end-point? What are the fields available for each data object?

METHODS
Which methods do I want to support for each object? Determine which CRUD (Create, Retrieve, Update, Delete). How about Search capabilities?

BROWSE
How will users browse files and/or data from the end-point?

DISCOVER CUSTOM DATA
Do I need to discover custom data fields and custom data objects from the end-points?

MAP
How will I map standard and custom data objects and fields to my applications data model? Will I give the user the ability to over-ride my default settings for standard objects? How will I treat custom objects and data and map that into my applications data model?

TRANSFORM
Do I need to transform any of the data structures? Are formats for Date, Time and other values consistent?

EVENTS & SYNCHRONIZATION
Are there events (e.g., changes to a data object) that my application needs to be alerted to?

OPERATIONS
Does my support and operations team need logging and usage data to support the integration? How will I handle alerts and notifications from the vendors regarding service outages, API changes, etc.?

STEP 5 Build and Release an MVP (Minimum Viable Product)

Now that you've defined a broad set of integration use cases, it's time to narrow the stories down into an iterative release plan so you can get your cooperative integration model to market and get feedback quickly. Some typical considerations for MVPs include:

Implement read-only use cases first. Delay the create, update and delete of user stories until later releases. Get the data moving between the apps first and then provide a means to update it later.

Initially include only the standard data objects and standard data fields from the end-point.

Provide a standard mapping of standard data. Provide the ability for the user to change your default data mapping and transformations in future releases.

Delay implementing more complex event handling and data synchronization scenarios until later releases.

Your clients want to interact with the other apps and cloud services they are using.

Don't inhibit them.
Help them by making your app cooperative.

cloud elements

Cloud Elements is a cloud API integration and aggregation service that developers use to integrate, monitor and maintain leading cloud services at a fraction of the cost and time. With their unique 'one-to-many' approach, a developer can integrate a single API to connect all the leading services in categories such as Documents, CRM, Finance and more. Cloud Elements is headquartered in Denver, CO, but serves customers worldwide.

More information can be found at www.cloud-elements.com.