WEBCON®

LOW-CODE, BUT BETTER.

AUTOMATION, INTEGRATION, WORKFLOW, AND PROCESS MANAGEMENT:

USING THE RIGHT TOOL FOR THE RIGHT TASK

WEBCON®
LOW-CODE, BUT BETTER.

It's a good time to be in the business process management/workflow/automation/integration business. Interest is on the rise. Microsoft is heavily promoting its Power Platform (which includes automation technology), robotic process automation (RPA) vendors like UIPath and Automation Anywhere are analyst darlings, and "bots" get regular treatment in the blogosphere.

But it's also a time wrought with confusion. Terms like "business process", "workflow", and "automation" are used interchangeably, and that invites apples-to-oranges comparisons between very different products and technologies.

Adding to the confusion are some pundits who rush to claim that a given tool can be used to solve all needs. In the Microsoft ecosystem, some members of the community that has formed around Power Automate (neé Microsoft Flow) have gone so far as to promote it for any given use case they can imagine.

One tool does not fit every task

Enthusiastic exploration of any tool is great, but with enthusiasm comes a desire to push boundaries to see what one can get away with. We see blog posts, conference presentations, and YouTube videos that are promoting the equivalent of:

- Driving Phillips head screws with a flat blade screwdriver
- Using a paring knife to pit cherries
- Using pliers to tighten bolts

Some people call this the Law of the Instrument. And it's what's happening with Microsoft Power Automate at the moment. Power Automate is a wonderful tool. I'm a fan. I use it personally (despite working for WEBCON) for some things because there are some tasks to which it's ideally suited.

You want an array of tools for an array of purposes

Much like, using the above metaphor, it will at some point become obvious that I should buy and use:

- A socket set
- A cherry pitter
- A cordless drill

I'm going to need process management tools like WEBCON BPS in addition to an automation tool like Power Automate.

The key is that they won't replace my screwdriver/pliers/knife — they'll complement them. It's all about using the right tool for the right task. And amassing a good set of tools in your toolbox so you can use the right one when the need arises.

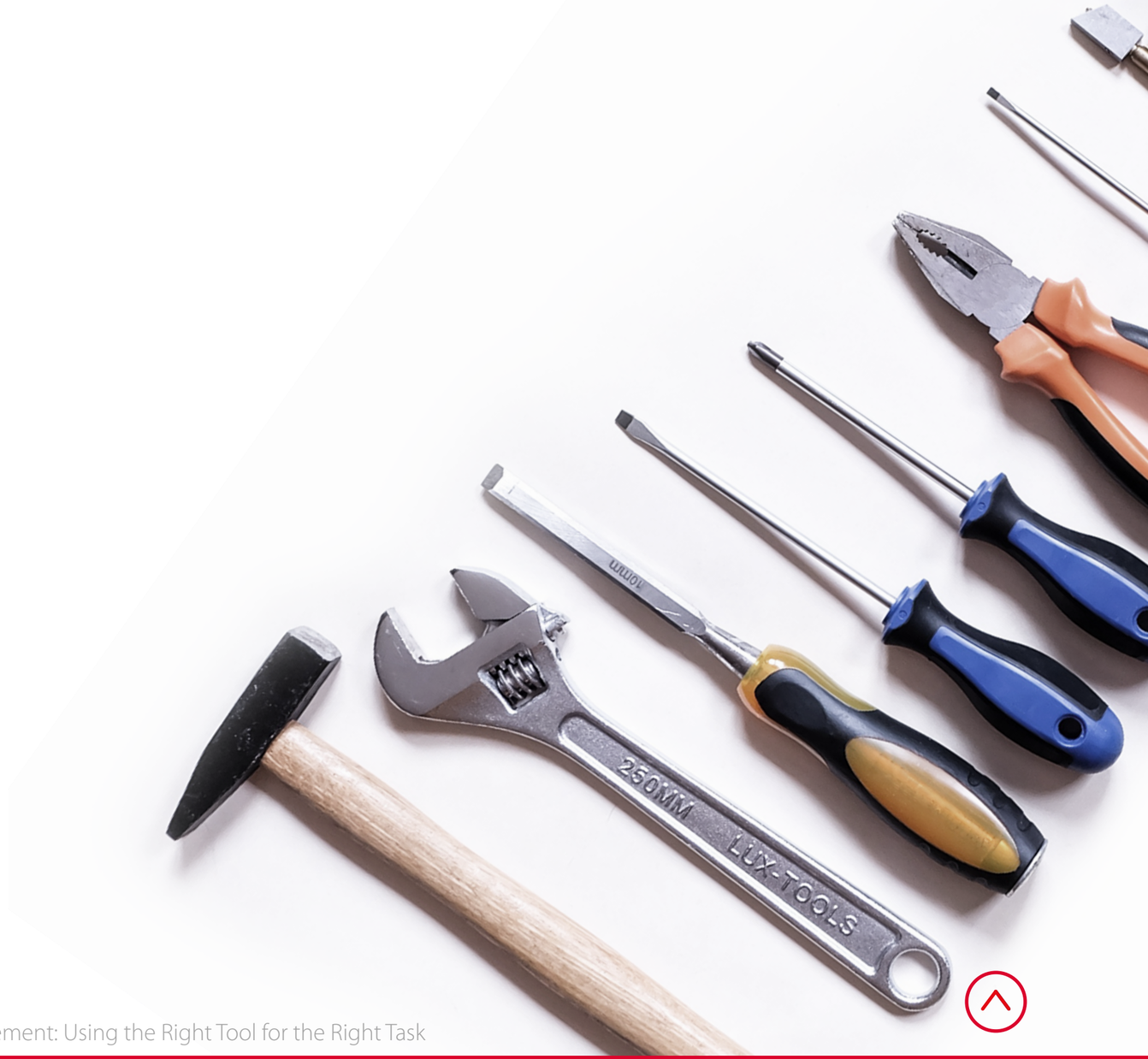
This is definitely true when we think about automation problems. No single tool fits every task. It's not a matter of simple vs. complex; it's about goodness of fit. It's about some problems warranting specialized capabilities.

You also want tools that address the project, not just individual tasks

I'll use a hammer to pound nails, a socket set to turn a variety of bolts, a saw to cut wood, etc. None of those tools, however, are designed to help me build a table. They can help with a number of tasks I need to perform while building a table, but the design and process itself? It's operating at a different level.

If you focus on task automation tools, you're often relegating the most important work (decisions, allocations, tracking, adjustments, design, etc.) to a "manual" process.
All too often, it's all done in your head. If you need to do this more than once, or if you need to achieve very specific and exacting goals, perhaps you'll invest in process tools, or at least project management tools.

And that's where we need to think about what we mean by business processes vs. automation. We'll cover that part last.



PROBLEM/SOLUTION

Automation Problem/Solution Types

Getting back to basic automation, it's important to note that there are many different kinds of automation problems. You can't use the same tools for all of them.

Let's look at a few scenarios. We could easily do more, but eight seems like a good number:

- Application automation
- Application integration
- Robotic process automation
- Data feed
- Data harvesting
- Data synchronization
- Document/form approval
- Content collaboration

Application Automation

The goal here is to automate the behavior of a specific application. This could be in reaction to an event, on a schedule, or when manually requested. Easy examples would include:

- When a document is uploaded to a OneDrive folder, change the filename to match the title.
- Once a day at 5:00pm, check the expiration date for all documents in a SharePoint library, and delete them if that day is today.
- Alert me via SMS text if I get an email from anyone of a specific list of sender email addresses.
- Save all email attachments to a specific OneDrive folder.
- Auto-provision Teams and configure them with specific settings and applications.
- Alert me via email to extremely good (or bad) scores in a Microsoft Forms survey.
- Add a row to an Excel worksheet with the values of a measurement taken every hour.

This category has been around a long time. Excel macros count. So do some PowerShell scripts. Definitely some Power Automate flows. And some uses of WEBCON BPS, Nintex Workflow, SharePoint Designer workflows, etc. would arguably be doing little more than this.

Several observations come to mind:

1. This is perfect for Power Automate.
2. It's very procedural. It's "when ___ happens, do _____, then _____, then _____". Flowchart-style modeling. If there's a visualization of what's going on, it's very procedural; "do this, now do that, now do that."
3. The goal is to eliminate the need to manually perform a series of tasks, not to eliminate those tasks or fundamentally change the work being done. And that's okay.
4. It involves linear logic. Once the set of instructions has completed, the automation is over. It's not really a long-running process of any kind, and while we might see some conditional branching logic, the automation is expected to move from a starting point to a finishing point with all possible haste.
5. It's the equivalent of what IT administrators write scripts to accomplish. And in some cases, you could accomplish a multi-page flow's work with 20 lines of PowerShell. Philosophically, it's the same thing.

A lot of work falls under this category. If you're not using Power Automate, I hope you're using IFTTT or Zapier at the very least. Performing these things manually is no fun.

What to use

The license for Power Automate that comes with an Office 365 subscription allows you to create flows that automate Office 365 assets, so it's a nice choice. Salesforce's process builder serves a similar function in that ecosystem. In truth, almost every environment of any real complexity offers some kind of automation tool.

Application Integration

I was tempted to make this just a special case of application automation, but there are enough issues to integration to warrant making it a dedicated category. Among other things, there are plenty of application automation tools that don't really care about integration.

In this scenario, you're trying to link two or more applications together automatically. Obvious scenarios would include eliminating the need to manually reentering data in more than one application, often taking the output of one application and using it as an input into another application.

As for examples:

- When a Microsoft Planner task is created, provision a GitHub project for it.
- When a tweet appears about your company on Twitter, send its text to Azure Cognitive Services to do a sentiment analysis on it. File the tweet and the analysisinto a SharePoint list.
- Add mobile photos to an Excel worksheet for expense reports.
- Use field data from a new SharePoint list item to generate a custom Word Document, then save it as a PDF file.

There's also a third option: pre-written connectors. It's a hybrid of both of the above options. It looks like option 1 in that it presents to the solution builder a catalog of sources it understands. That said, the connections to those systems are nevertheless standards-based. The goal is not so much to free the builders from juggling proprietary APIs, but rather to free them from the need to know how to use REST at all.

The key to many integration/automation tools, certainly in the case of Power Automate/IFTTT/Zapier, is a library of pre-written connectors for a large number of diverse applications and data sources. Some connectors merely read/write data, whereas others can ask those applications to carry out operations. Some even support receiving alerts when an event takes place in those external applications/data sources.

Again, a number of observations come to mind:

1. Again, this is perfect for, and in fact the design goal, for Power Automate (as well as IFTTT and Zapier). At least at first. Keep reading.
2. This is conceptually the same thing as Application Automation. We're eliminating manual steps, and hence saving time, increasing speed, reducing errors, etc. We're not, however, attempting to change the nature of the work itself.
3. Because one is not changing the work itself but merely automating it, this means less design time and quicker initial delivery.
4. The utility of this is a direct function of the quality of the breadth of the connector library (how many applications/data sources can it reach) and the depth/quality of the individual connectors (which API calls and methods can be invoked; are key things supported or missing).

That said, there are also some caveats…

Taking connectors on faith can backfire

Stringing together a chain of requests and passing data between them requires a lot of faith that each request will succeed, that its results will remain consistent, and a lot more. Most initial efforts make a leap of faith that nothing will go unexpectedly (let alone wrong). In order to make this scenario go well, you need a lot of exception handling and/or proactive state checking to make sure the target system is ready to handle your request.

For example, a connector to a to-do list system may support creating new to-do items, but not checking to see if they're complete, reacting when they're finished, or alerting anyone when they become overdue

This is not a bad thing – unless you were expecting prewritten connectors to be "magic" (because nothing is magic).

APIs aren't always stable over time

At the time of this writing, a number of vendors' pre-written connectors to Facebook have stopped working because Facebook made changes to their API. Some of the vendors are reacting quickly, but not all of them, and even the ones that are doing so don't have a uniform sense of urgency.

Prewritten connectors leave things out

Virtually all prewritten connectors do not implement 100% of the APIs exposed by their target applications/data sources. That's by design; if they did, they'd be as complicated as those APIs. The deeper the need you have to integrate with a service, the greater the likelihood that the prewritten connector is missing a function you need.

That means you'll be writing or configuring some of your own custom connectors, or at least making one-off calls to remote APIs. Again, this is not bad – unless you were expecting magic.

Prewritten connectors might not be permitted

A prewritten connector, or even a custom connector, is valueless without the necessary permission to use it.

The focus on connecting data tends to ignore issues of privileges and co-operation. The most often-cited examples tend to involve systems to which one has direct ownership (e.g., my calendars, my customers, my emails, my document repositories) as opposed to those governed by different departments.

When you span departments, you need to start thinking about permission to data. Will sales let marketing connect to their CRM system and do whatever they want with it? No, and marketing shouldn't expect them to; there are licensing and training issues even if everyone trusts everyone. Will the team that supports SAP allow some SharePoint developers to make direct connections to their environment without any involvement on their part? I doubt it. An integration strategy cannot be based on a desire to bypass.

Take heart - seriously

Essentially, integration is almost always harder than it looks. As long as you don't expect magic, you'll be pleased with how much easier it is today than it was even two years ago.

What to use

If you want to integrate using pre-written connectors, Power Automate, Zapier, IFTTT are good options. Workflow automation products, like WEBCON BPS, Nintex, or K2 usually offer a combination of a (smaller-scale) library of prewritten connectors, a set of general-purpose connection builder tools that offer the ability to save and reuse said work, and APIs that allow to pull and push data from and to external systems. They often also offer web APIs to allow external systems to interact with their workflows.

Not every solution builder needs to become a REST expert and a master of various vendor APIs. If the workflow software you use allows connections to be saved and reused, you're creating your own connector library of sorts.

And the best option of all, particularly if you're dealing with different departments controlling different data sources, you have Department A built a reusable workflow (e.g., "Post a new sales lead") that can be invoked by a workflow written by Department B (e.g., "record a conversation at a booth exhibit"). As such, you're integrating with processes rather than with data access.

And as it turns out, APIs aren't the only way to do application automation/integration…

Robotic Process Automation

RPA has gotten a lot of attention in recent times. Microsoft just announced that Power Automate will also be able to create what they're calling UI flows. UIPath has become a very popular company, as have Automation Anywhere and Blue Prism. Many workflow vendors are now adding RPA as an option through acquisition, partnering, or new development efforts.

RPA involves the generation of bots (hence the word robotic in the name), automated web browser or desktop sessions that connect to applications, simulate a user typing and clicking, and read the contents of what is being displayed. A good RPA environment knows how to handle popup messages, OS exception conditions, and much more.

Why do this? Two main reasons:

1. You don't have an API you can use to interact with an application, or the API that does exist is terrible.
2. You lack the expertise to deal with the nuances of using that API, but you do have the ability to record interactive screen sessions and indicate which information should be read/written automatically.
3. Your IT department, system integrator, or LOB software owner is not willing to, or not able to cooperate on your project – at least not right away.

RPA lets you record manual interactions with software, indicate what could change based on inputs, and what needs to be read off of the screen and passed on.

There are at least three caveats to RPA:

1. If the application is updated and its user interface changes, your bot becomes invalid.
2. Every bot session is an automated user session; either a browser is being instantiated or a desktop application is. This can get very resource intensive at scale.
3. Individual records have unique IDs; while database queries and function calls use them, they aren't always displayed on the screen. Even when the master item's

ID is visible, it's rare to see visible IDs for line items or linked data (e.g., a purchase request's equipment list and preferred vendor info. And if an ID isn't visible, RPA can't screen-scrape it. Mitigating this by simulating an interactive search carries risks.

Despite these above three issues, there's still a temptation to overuse RPA; to a non-developer who thinks REST means relaxation and SOAP is for washing hands, the idea of recording screen sessions is likely to be appealing

Even though it's called Robotic Process Automation, it's not really about processes

Actually, the biggest caveat is the name "robotic process automation" – because what's being automated are tasks, not processes. The term should be "RTA" as opposed to RPA.

We'll get to business process management in a little while, but much like application automation or (API-based) application integration, RPA automations are not really for automating an entire business process by themselves.

They do, however, make excellent actions to be invoked by a true automated/managed business process.

What to use

UIPath has a lot of experience with this, and they're offering more and more API-based integration options as well. Automation Anywhere and Blue Prism are the other two "usual suspects" when considering multi-purpose RPA vendors. Microsoft's introduction of UI flows into the premium tier for Power Automate bear examination; they have a lot of promise, although no (at the time of this writing) proven experience. And integration, especially UI-based integration, is always harder than you think it is.

Data Handling

So far, we've looked at task automation and sending instructions to one or more applications, perhaps transferring a bit of data in/out. Now let's turn to data-centric activity, and how to automate it. There's no shortage of options for doing so. Moreover, as you might suspect, there is a faction that loves to promote doing this with Power Automate whenever possible.

Data transfer is more nuanced than one might think, though, and in this case, the nuances really matter. Let's look at three kinds of data handling tasks.

Data Feed

Consider a public-facing (likely anonymous) Microsoft Form or custom web page that collects event registrations. It's quite likely that the entered data is being collected in one place (site, server) and needs to be sent somewhere else (e.g., a SharePoint list or a database table) to actually do something with it (analyze it, use it in other processes, etc.).

Characteristics in play here are:

1. It's a trickle. There's no reason not to send the data from the collection point to the true destination point immediately if possible. We don't need to wait to send it in batches.
2. The original data becomes unimportant. Once transferred, the source data at the collection point is not important. It can be deleted without any adverse effects.
3. No dependencies. Data being sent from one entry doesn't depend, contribute to, or become aggregated with, data from another entry until they've been sent to their destination.

In other words, the flow of data is one-way, it's in quick single-item bursts, and there isn't a lot of transformation going on prior to it arriving at its destination.

What to use

Almost any tool can do this, and the straightforward react-and-act approach used by Power Automate, Zapier, IFTTT, etc. is perfectly fine, especially since they already excel at making connections.

That, said mechanisms must be in place to ensure that every entry made in the source system absolutely moves to the target. There must be an error raised if that fails, and there must be a way of detecting which items didn't get transferred when something is interrupted.

If you require more assurance that no data row will be left behind, we come to…

Data Harvesting

There are data collect-and-send operations that are not simple, however. There might be a lot of pre- and post-processing transformation taking place on the data being send from source to destination. Dealing with periodicity in the data might require waiting until the end of the period to send it as a batch, computing totals and other statistics when doing so.

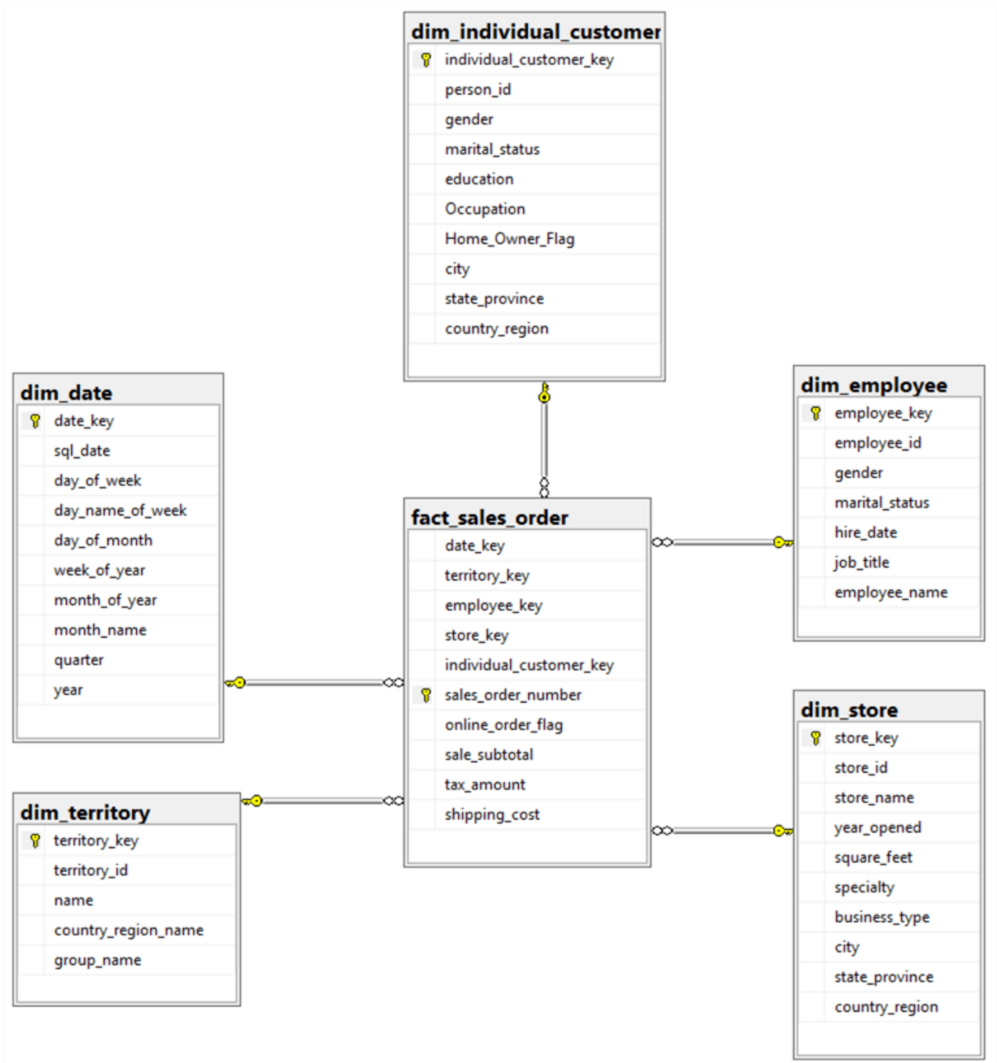
In fact, it's necessary to do this in batches. If the data changes minute-to-minute, two reports done a few minutes will yield different results, and this will be a source of organizational confusion.

Data harvested for decision-making and analysis purposes needs to be in a stable state, at least for defined periods of time.

Enter tools that are often categorized as Extract/Transform/Load, or ETL for short. There are a lot of them, and you should pick one and use it.

Consider scenarios where the source is a heavily normalized operational database and the target is a business intelligence-optimized data warehouse. In such cases, the schemas look very different, and Transformation become even more important than Extract and Load.

Data Harvesting could be used to extract data from an operational database such as Microsoft's AdventureWorks Sample Database on the left and create/populate/ add to the star schema on the right, which is far more optimized for business intelligence queries.



What to use

ETL tools know how to compute hourly aggregates, precalculate values, etc. and they know how to do it at scale. Workflow engines aren't optimized to work with large volumes of data.

You could use a general-purpose automation tool like Power Automate for this, but you'd wind up with:

- A lot of extra logic to handle exceptions.
- A lot of calls out to Azure functions and other places to handle custom transformations.
- A lot of extra logic to handle logging and tracking.
- A lot of extra work to deal with calculated totals and other custom operations.

ETL tools have that stuff built in. And they're designed to handle large batches with high performance.

If you're a SQL Server customer, SQL Server Integration Services (SSIS) remains an excellent tool for ETL. Azure customers should regard Azure Data Factory should be the ETL tool of choice. If you've become familiar and comfortable with SSIS in the past, Azure Data Factory can even run SSIS packages.

Data Bricks, an open source approach to working with data sets, include ETL functionality, and Microsoft Azure has an implementation of Data Bricks as well. As do other cloud companies.

In some cases, data harvesting can be an interim step along the way to data harvesting; a Power Automate flow (or any automation tool) could feed data into Azure Event Hub, a queue mechanism that can be treated as a data source, and thus used for harvesting purposes. If it isn't necessary to build a star schema, one can query it directly using Azure Streaming Analytics.

In truth, almost every data management environment provides some ETL facility — and they're all superior to pressing a general-purpose automation tool into service for this purpose.

Data Synchronization

Disturbingly, far too many of the examples provided for Power Automate involve synchronization of data between systems (e.g., Google Calendar with Outlook in Office 365) – a task to which it, and any tool focused on general-purpose task and/or process automation, is ill-suited.

Such examples usually take shortcuts, like:

- Omitting what to do when a contact is deleted.
- Only allowing changes to happen in one of the two places, so it's not really a sync, more of a one-way copy.
- If you're synching contacts, using the contact name or email address as a unique identifier.

In the real world, conversely:

- Names undergo spelling corrections.
- Email addresses change, and so on.
- Deletions happen.
- Changes get made in more than one place.

Synchronization is a data management problem, not a business process problem. It's difficult. It may involve keeping and comparing copies of some of the data to check for deletions, being aware of the unique ID scheme behind multiple data stores, how to monitor different systems' change and deletion logs, how to handle repeating patterns, and more.

What to use

Synchronization is not simple. It's why there are dedicated synchronization products on the market. They exist for a reason. They range from cloud services to mobile apps, costing anywhere from $6.00 USD per month to thousands of dollars for a one-time purchase. Regardless, if you need this, they're worth every penny.

Data Routing

It's at this stage that we start to get into what feels less like the automation of repetitive tasks and more like the beginning of actual business processes. There are people involved, tasks assigned to them, assets involved, and so on.

It's still more of a matter of automating the routing of the content than it is transforming what the actual business goals and strategies/tactics to achieve them are, though. So it stays in the "types of automation" taxonomy for the purposes of this paper.

Not surprisingly, there are nuances. Let's look at three types of content routing…

Document/Form Approval

Document approval is an extremely common use case. One (perhaps more than one) piece of content (a Word document, a web page, etc.) needs to be approved before something else can be done with it (printing, publishing, archiving, emailing, etc.).

There may be nothing there to facilitate doing anything to the content being passed around. There may be a way to provide comments. The decision presented to the reviewer might have more options than Approve and Reject, but the model remains (1) show the reviewers something, (2) have each of them make a choice.

Not surprisingly, this facility exists practically everywhere. It's not a difficult pattern to implement.

Form approval is a special case of document approval. From the point of view of logic flow, there is no difference. The only real difference is that the content is structured into a form, and that means the solution will require a form design and rendering tool and/or custom page development. It also allows some of the form fields to be used for notification messages, automated UI behavior, etc.

But it's still approving a piece of content and either rejecting it or moving it along.

What to use

You can use almost anything. Certainly Power Automate works. The decisions you're collecting are single-valued (accept/reject, yes/no/discuss, etc.) and some outcomes end the flow completely. Zapier has some limited support for these scenarios Obviously business process tools like WEBCON BPS, K2, Nintex, or the others will do as well (among other things).

The issue is that the real world is not as simple as the above tools suggest. For example:

1. Participants might want to modify the contents of the form (or document) while they are reviewing it.
2. Participants might not want to end the process when it's rejected, but rather return it to the originator to have changes made, after which they're again submitted for review.
3. There might be several forms, or parts of a form, that need to be completed by different people. As such, there's no one "thing" to "approve".

Those issues lead us toward not content approval, but rather content collaboration. At which point Power Automate's options for handling the above three issues has real limits that you'd need to address via a lot of custom work of your own or by using a workflow and/or business process management tool (WEBCON BPS, Nintex, K2, Bizagi, Appian, Pega, …).

Content Collaboration

In this scenario, we allow for the workflow's assets (the documents, the forms, etc.) to evolve as the process progresses. We aren't just routing something around for decisions; we're routing it around to get it completed.

Some of the steps in a form (or document, or page)'s process will involve adding new content (in fields or in free form text) to the case. Other steps will involve approvals, but still other steps won't be so much to provide official approval as they would be to solicit reviews. Some responses have an effect on others. Some responses will trigger a requirement for additional form sections to be completed or additional documents to be written and added to the case being passed around. Some will cause content to be automatically created.

The workflow can determine which steps are necessary and who should perform them. Some steps will be approvals, but others will request additional input. Some responses will necessitate different steps, and so on.

We've seen many, many projects that began life as document/form routing evolve into content collaboration patterns. It turned out that users had to provide more than an approval/rejection. Ultimately, "approval" turns out to be too limited a concept.

What to use

Power Automate is not ready for this; not without a lot of work on your part. You'd need to create a companion PowerApp to go with it and hand-code a way to keep them linked together. For example:

1. Saving the PowerApp the first time starts a flow.
2. The flow changes a value in the data to which the PowerApp is bound.
3. When you open the PowerApp, it looks at that field and executes formulas to modify the fields (or selects a custom screen) that corresponds to where the workflow is.
4. When the user clicks a button in the PowerApp, it changes that value again, saves the data, and exits.
5. The flow looks at that value and branches accordingly.
6. Unless the process is finished, go back to step 3.

That's a lot of customization to maintain. Moreover, it doesn't address recording and making usable a coherent audit trail, or locking the form down at different points in the process so only certain people can do certain things to it.

It essentially means that some of your logic will be in the logic tier and some of it will be in your UI tier. It will be hard to maintain and harder still to explain.

Unless you're using Power Automate's business process flows in combination with a model-driven PowerApp, both of which require that data be stored in the Common Data Service, this really should be out of scope for that platform. Zealots will try to juggle the toolset in a manner similar to what I just described, but it's at this point that you genuinely need to look at the tools provided by WEBCON or one of the other third-party options.

BUSINESS PROCESS

Business Process Management

When *what* should be done and *who/what* is needed to accomplish it is as important as (if not more than) *how* it should be done, you're starting to think about a business process.

While the terms "automation", "workflow", and "business process management" are often used interchangeably, let's be a bit more thoughtful about it. An imperfect but useful way to view them could be:

- Automation: the execution of tasks with limited manual intervention or without any manual intervention at all.
- Workflow: a progression of activity from start to finish along a predetermined path. Not every step need be automated, but many if not most indeed are.
- Business Process: everything needed to complete a business goal. This could include one or more workflows, forms, documents, roles, people, tasks, resources, and other assets.

Tools intended principally for automation tend to focus on the activity, the details (the how) rather than the overall goals and steps (the what and why).

As a result, people building business processes using automation tools tend to rely on an additional product (often Visio) to document and explain what the workflow does, and still other disconnected tools to model/store data and analyze process metrics.

Business Process tools tend to put the goals and steps front and center; implementation details are part of the model but are kept behind the scenes to avoid clutter. It stems from the following mindset:

- Business process management: "let's do things better, perhaps even do better things"
- Automation: "let's not do things by hand"

When you want to be able to choose whether to evaluate (and explain) what is being done (with an eye toward improving it) or whether to drill down on how steps are implemented, you're doing business process work.

In this type of situation, the business process is responsible for the decisions, the metrics, the monitoring, the allocations, the assignments, etc. They invoke automations that take care of the activity. Some people call this orchestration.

An example

Consider employee onboarding. Really consider it. The whole thing. Not just the account creation part done by IT. Everything:

- Extending an offer.
- Negotiating a salary.
- Determining a start date.
- Allocating office space.
- Allocating parking privileges.
- Generating security passes and keys.
- Procuring equipment.
- Creating network credentials.
- Allocating licenses for a variety of software that supports single sign-on.
- Creating usernames/passwords for software that doesn't support single sign-on.
- Attending orientation.
- Explaining expense reporting.
- Explaining travel procedures.
- Registering benefits package choices.
- Setting up payroll and direct deposit.
- Scheduling a review at the end of the initial probationary period.

…and we're not done. But that bulleted list? That's the majority of the employee onboarding business process. Any one of those bullets could be an application automation (whether or not it involves integration – and of course it does). But the whole thing is geared toward a common strategic goal: bring on a new hire with the happiest experience, the least effort, at the most efficient pace, with the least ambiguity and fewest errors. Happy employee, happy manager, happy company.

If one wants to manage the company's relationship with an employee from recruitment (or better still, from the creation of the open position) to termination, it gets more interesting, handling reviews, promotions, etc., and one process might have subprocesses, each of which might call out to automations to perform actions.

What to use

The answer for this one is a little more involved.

Power Automate alone won't be enough. A case could be made that the combination of Power Automate, PowerApps, the Common Data Service, some Azure Functions, and maybe even some Power BI analytics could be brought together to create a business process, but the glue between them is in the head of the designers and any hand-created documentation they see fit to create. To their credit, Microsoft Dynamics (which owns the Power Platform) has worked on making it possible to deploy most of these assets as a solution set. Maintaining them as a set is quite another matter.
They're really focused on a data-centric approach to business problems, where a PowerApp is the face of a solution, CDS is its heart and soul, and Power Automate is a vehicle for operating on or reacting to the data. A lot of business cases need exactly this approach -- just not process-centric solutions.

In process-centric solutions, the process is front-and-center. The data is determined by the process. The user interface and analytics are determined by both of them.

Many platforms have this in mind. WEBCON, obviously, but a process-centric approach is also practiced by Appian, Bizagi, Pega, and many other vendors. Nintex and K2 lean toward process-centrism, but they still require data to be finalized before an application can even be started, so you have to create your data (and do so outside of your solution) before you know what you're going to do with it.

There are a couple of types of business processes that merit specific attention…

Case Management

Case management is a particular approach to managing business processes, particularly when the way a process will be carried out varies widely depending on the situation. It represents the union of content management and process management and often task automation.

Central to the approach is a *case file* (or, if you prefer, a case folder), a repository and work area for all of the assets relevant to achieving a goal (curing a patient, representing a defendant, investigating a crime, etc.). That includes documents, images, data, applications, a library of automated options, and more.

The other nuance to case management is that the models are less focused on prescribing what *must* happen. Instead, the goal is to model what *can* happen, the possible paths a case can take. It's the job of the *participants* to decide which paths will be taken, as well as who should participate.

It's the job of the tool to make sure that case files contain the right tools and content, and to keep track of everything that happens for versioning, auditing, and reporting purposes. To harvest data from the case and put it in places on which it can be reported/analyzed.

It's the job of the designer to consider which paths the case might traverse and provide a library of automations and actions to assist them.

If it's not obvious by now, case management is about flexibility, adaptability, and embracing change. It's also about preserving an audit trail of everything that happened and who did it, and making the audit trail itself into a case file asset.

What to use

If it weren't for some of the auditing and resource requirements, workspaces provided by Microsoft Teams or SharePoint (Server or Online) could serve as case files, and a number of Power Automation flows and PowerApps could be associated with each site.

It sounds nice until you realize that you'd need to allocate one team (or site) to each case instance. It would be a challenge both in terms of organization and resources. But conceptually, it fits.

That said, a number of business process/workflow companies are either optimized for case management (e.g., WEBCON) or offer case management as a module (e.g., K2). One way or another, they implement case files (mini-workspaces) without the above issues.

Task Management

As mentioned earlier, approvals are only one kind of task. We crossed the chasm from approvals to contributions+decisions when we shifted from document/form approvals to content collaboration. And all tasks are not created equal.

- Basic tasks have names/titles, often have descriptions, assignees, and can have due dates and reminders. They can be tied to a time or even a place.

  But the actual work to be performed has to happen elsewhere. The tasks refer to the work, nothing more. You'll follow a URL to a page to do the real work, then return to the task to mark it complete.

  Task management systems operating like this include Microsoft To-Do, ToDoist, Apple Reminders, and more. Such environments are usually concerned only with assigning and completing tasks, perhaps monitoring them as well.

- A workflow task can do more than this. A task can have its own flexible user interface which includes presenting custom information and in turn collecting user input.

  These kinds of tasks are the work; by combining the task metadata with the task itself, users need only work in one place.

  Examples of this would include SharePoint task lists (different tasks based on different content types) and Exchange tasks visualized in Outlook (different tasks based on difference message types). When being created, they could be populated with custom data, and upon completion, they could be queried for new user input. Regrettably, neither approach is fashionable any longer at the time of this writing.

  Power Automate supports approvals, which can have some information supplied as a notification message, and for which a single decision can be recorded, but it's a very limited scenario.

  Fortunately, every workflow and business process tool suite I've ever seen covers this approach to tasks. Even SharePoint Designer + InfoPath handled it well. But if you're working with tasks

Task Orchestration

A key part of task management (that not every toolset/platform/environment supports) is coordinated subtasks. In those circumstances, tasks can in turn have subtasks associated with them, and subtasks are essentially subordinate linked workflows in their own right; it's the duty of the tool to support the delegation and roll-up logic.

Consider a review process that has a proposal routed from department head to department head for approval/rejection, but within each department it is given to team members for advice and comment. At the completion of each departmental subtasks, the department heads act on the primary tasks.

That's one large process with a master workflow moving from department to department, with subworkflows happening within each department. Tasks are assigned to department heads from the parent workflow and to team members in the child workflows. When it comes time to audit the process, though, it needs to be viewable as a single thing.

You should not expect to see elaborate task handling in a straightforward automation product like Power Automate. You'll find some of the functionality in a good workflow product and pretty much all the functionality you'd want in a business process suite. Yes, you could write all of that parent/child and roll-up logic yourself. But you could also simply use a toolset that understands and supports this in the box.

CONCLUSION

Conclusion

Many of these process flow patterns tend to be evolved versions of other patterns. It'd be perfectly reasonable to see Application Integration as an evolved version of Application Automation. Similarly, Data Harvesting is an evolved case of a Data Feed. Content Collaboration is what many Document or Form Approval patterns become over time. And over still more time, collaborative creation becomes Case Management.

The reason for bothering with such distinctions is to help decide how to approach a given problem and with which tools. For example:

- Any automation tool will work for document approvals. Any workflow tool combined with a form tool can handle form approvals. Collaborative content creation requires logic more akin to that of an integrated business process toolset, especially since forms and/or documents may be sent back and forth between steps until everyone is satisfied.
- Any automation tool with connectors can pipe data from one place to another. Keeping data in synchronization between systems is an ongoing data management process. Staging data harvested from one application into a completely different form for a different application is a data-intensive business process. Specialized tools are a better bet in those cases.
- Any automation tool with connectors can integrate two or more applications, but depending on who controls those applications and which policies are in place, as well as which skill sets, you may need an automation toolset that allows the creation of components, reuse, delegated privileges, and so on.
- Any automation tool can improve the efficiency of a set of activity (more speed/fewer errors), but when you want to improve the efficacy of that activity (business impact), you'll cross the chasm from automation to business process management. Requirements for any degree of flexibility will likely trigger a heightened interest in case management approaches.

In the above bullet points, you can replace "any automation tool" with "Power Automate", and as automation tools go, it's *very* good.

That said, if you move from automated activity to business process management. Power Automate won't be enough. You'll be faced with two choices:

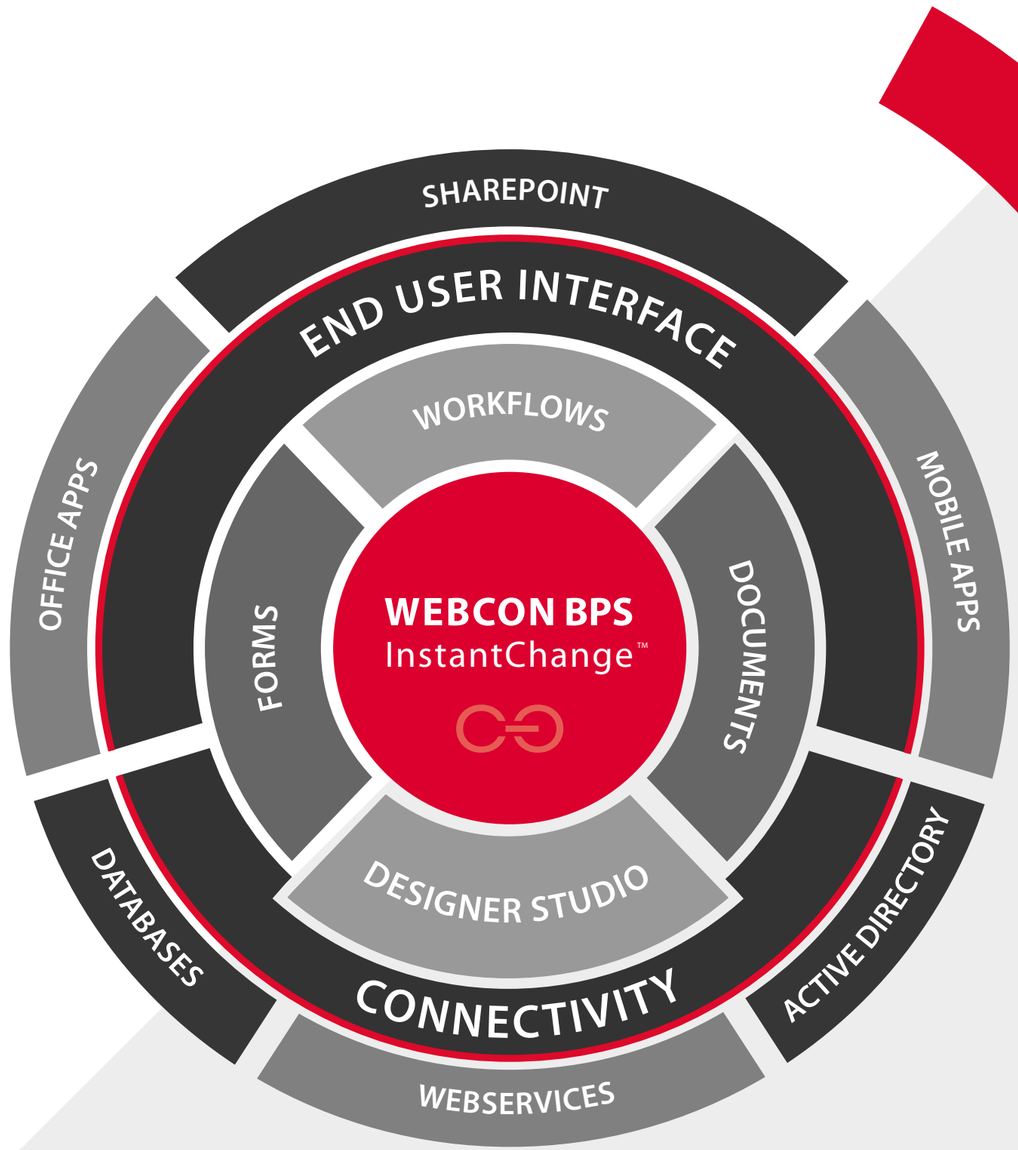
1. Supplement the Power Platform with a lot of extra manual work. You won't need to code like a developer, but you'll still need to think like one. You become the business process management environment, juggling a lot of discrete pieces like flows, PowerApps, Azure functions, Logic Apps, and more. And keeping them in sync. People who advocate this approach, not coincidentally, are developers.
2. Look at a business process management tool.

Where WEBCON fits in

Until this final section, this whitepaper has been as neutral and agnostic as possible. You could ask other workflow ISVs, and even (on a good day) Power Platform people to read it and they'd happily (or at least reluctantly) nod their heads.

However, for a few closing paragraphs, it's time to get completely WEBCON BPS-specific…

- WEBCON BPS is equally suited to content routing and content collaboration, both forms and documents. And if you need the input of multiple people, the approach it takes to having the workflow modify the form as it moves from step to step makes it genuinely superior to the way other platforms work.
- WEBCON BPS's approach to both state machine-based process design and case file-based data management makes it superior for case management-based solutions. That's probably worthy of a whitepaper of its own.
- It's approach to registering, securing, and reusing connections, data sources, and subprocesses make for a superior experience when it comes to integration between assets curated by different parts of your organization. If you do need to create a connection "manually," you can take comfort in knowing you'll be able to reuse it over and over.



- WEBCON BPS is excellent as an automation/integration tool. When it comes to integrating with APIs, it has support for SQL Server, Oracle, SharePoint lists, SOAP web services, and RESTful web services. The moment a prewritten connector lacks a key feature, WEBCON BPS becomes at least as good as anything else at API-based integration. What's more, it allows for very elaborate logic to be applied to those connections and their contents.
- WEBCON's integration approach (as well as its application design approach) is based on configuration effort, not coding/scripting. That said, when you need to use code (proprietary APIs, for example), the platform's .NET SDK is there.
- Because it's a business process tool, issues like long-running workflows are not an issue. Neither are processes that move back-and-forth. Because it's a business process tool, it can focus on the overall efficacy of the process to meet business goals – and then drill down into the implementation details.
- If your need to show or explain your application is important, WEBCON BPS diagrams are really easy to read. It has neither the clutter of procedural products/platforms nor the academic abstraction of traditional BPMN-based approaches.

Where wouldn't you use WEBCON BPS?

- WEBCON can certainly funnel data, an item at a time, between two points, but when it comes to data harvesting or synchronization, again, that's a job for ETL or dedicated sync tools/platforms. Nobody's workflow or business process or app automation tool is the right choice.
- If you really need a series of prewritten connectors, and the ones you'll be using aren't missing key features, and your automation logic is very straightforward, linear, and short-running – we'll be the first ones to suggest Power Automate or something similar.

Essentially, if your interest is in automating nothing but Office 365 assets and the logic you have in mind is short-running and one-directional, Power Automate won't cost you extra to use, and that's a compelling motivator.

The moment you need to talk to non-Office 365 resources, or your logic needs become elaborate, your "in the box" Power Automate/Power Apps license won't cover that. The Power Platform's premium licensing offers no advantage over the breadth of ISV-authored tools for integration, automation, and business process management. Not surprisingly, we'd hope you look at WEBCON BPS.

WEBCON®
LOW-CODE, BUT BETTER.

THANKS FOR READING

Thanks for reading!

We meant every word, and we took great pains to avoid exaggeration. That said, we'd rather you didn't have to believe us – not when you could believe your own eyes.

Check out webcon.com, where you can find documents, videos, testimonials, case studies, and an online trial.

There's even a downloadable Express edition of our product you can install and use for free – in production – and never expires.

There's also a link to contact us and arrange to see a demo of WEBCON BPS in action -- which is even more fun than reading this document (but again, thanks for doing that).

We look forward to your virtual visit.

Let's talk!

To learn more visit our website: webcon.com

+1 206 513 2513
office@webcon.com

About the author

Mike Fitzmaurice is the Chief Evangelist and VP – North America for **WEBCON** and is a recognised thought leader in workflow/business process automation, citizen development, and low-code/no-code solution platforms and strategies. He has more than 25 years of product, consulting, evangelism, engineering, and IT management expertise. His 11 years at Microsoft included being the original technical product manager for SharePoint, helping launch and shepherd its first three releases; it was Mike who birthed, developed, and led developer evangelism on SharePoint, positioning it as a development platform. Other gigs included a decade at Nintex as Vice-President of Workflow Technology, a year as Chief Technology Officer for skybow, and five years as Director of IT at the National Association of Broadcasters.