5 KEYS TO UNLOCKING A COMPLEX CITYWORKS INTEGRATION

BY MILES KELLY AND VICTOR STAGGS, WOOLPERT

There's no question that building integrations to automatically connect Cityworks with other enterprise systems can provide a huge return on investment. However, ensuring the success of those integrations requires considering their wide range in complexity, from the simple one-way nightly data sync to the more complex near-real-time bi-directional integration.

Before getting started on your next integration project, here are a few considerations that can help you set the project requirements accordingly.

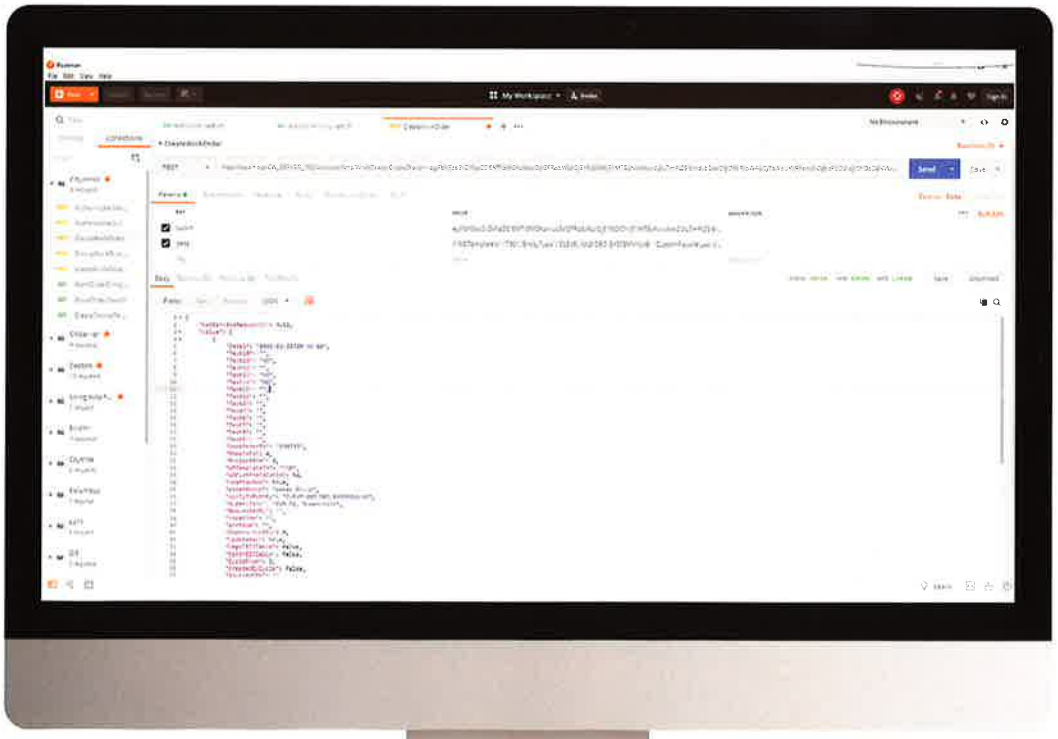
CLEARLY DEFINE YOUR END GOALS

Importing employee data is a common example of a simple integration. If human resources software is the system of record for employee data, users would naturally add and update employees in that system. An integration might push data such as name, title, department, and hourly rate into Cityworks on a regular basis.

While manually updating employees in Cityworks may not take a lot of work, an integration will eliminate problems like having Miles Wight-Kelly in one system and Miles W. Kelley in the other. The end goals, in this case, may be as much about achieving solid data integrity as reducing staff workload.

EVALUATE API CAPABILITIES

The most important tool available for building integrations is the Cityworks application programming interfaces (APIs). The APIs allow data to be sent to and from Cityworks through automated HTTP calls, instead of having to enter data manually. With every new release, Cityworks expands the capabilities of its APIs. For example, basic functions such as getting a list of employees have been around for a while, but as of Cityworks 15.2, employees can be added and updated through the API—exactly the functions necessary in this example.



Postman is a free application programming interface (API) development tool that lets users make API calls and investigate the responses.

The API documentation available on MyCityworks shows which methods require additional licensing and walks users through authenticating with Cityworks and making calls to the API. Free, open-source tools such as Postman make it easy to play with APIs, test out how to make calls correctly, and view the responses from Cityworks.

UNDERSTAND ERROR TOLERANCE

One aspect critical to architecting an integration is understanding a team's tolerance for errors, which are inevitable when dealing with distributed systems. Consider the risks involved if the integration fails.

In this employee data example, if the system imports employee data to Cityworks Monday night, then it fails Tuesday night, but runs successfully again on Wednesday night, what are the consequences? During the day on Wednesday, that Cityworks data would be a day old. But Wednesday night it would be corrected, so by Thursday it would be up to date again. The data updates would be delayed, but not lost.

What are the implications in this case? Would a supervisor need to add that new hire to the work order on Wednesday, or could it be completed the next day? If an employee received a raise and it didn't show up in Cityworks until Thursday, is the organization comfortable with the work order cost entered on Wednesday being slightly low, or is a higher level of accuracy required? These are all important questions to document and consider.

INFORM THE SYSTEM ADMINISTRATOR

Even if an organization can accept some minor consequences of an integration failing occasionally, the system administrator still must be made aware every time it fails—before questions are asked about why that new hire hasn't shown up in the system yet. The integration should raise active notification of error-level events. One simple approach to this problem could be to use one of several open-source logging frameworks, such as nlog or log4j/log4net/log4javascript, that make it easy to send error alerts by email.

IDENTIFY COMPLEX REQUIREMENTS

If there is a need for higher reliability, or the requirements are more complex—bi-directional integrations or dealing with transactional data, for example—organizations may need to consider a more sophisticated architecture, using tools such as Cityworks WebHooks, or even a message-based architecture and Enterprise Service Bus (ESB).

In many cases, organizations with IT staff who have the skills and availability to create an integration in-house can take advantage of of Cityworks APIs along with free tools and frameworks. Identifying the potential risks and planning the architecture accordingly will allow organizations to take full advantage of the potential value in sharing data across the enterprise.

Miles Kelly and Victor Staggs are application developers at Woolpert. Contact them at miles.kelly@woolpert.com and victor.staggs@woolpert.com.

How many external systems are there?
Simple integrations may only have a single external system. Or there could be several integrations between Cityworks and multiple external systems.

Is it easy to work with the external system?
In a perfect world, all external systems are modern and up-to-date and have robust, mature, modern APIs. In reality, integrating older systems with rudimentary or non-existent APIs can be challenging.

How complex are the business processes and rules?
Generally, more complex business processes need a more complex integration. However, this also presents an opportunity to refine your workflows for better outcomes.

How many dataflows are there, and what types?
Document the workflows and the data that inform them. In a simple one-way dataflow, data travel from an external system into Cityworks, or vice versa. A bi-directional data flow often requires that new records, changes, or cancellations be reflected in both systems—increasing the complexity of the integration.

How quickly does data need to move from one system to the other? Does delivery need to be guaranteed?
For some integrations, it may not be necessary to reflect changes in one system immediately in the other system. In other integrations, like event-driven integrations, a notification to an external system is needed as soon as a work order is closed.