Equipping Yourself For Designing and Expanding Skills and IDWs

Imagine you're halfway up the hill (Everest) and you come to a vertical ice wall. You might be able to scramble up it with some ropes and crampons, but your whole team could get up it much faster with an ice pick specially designed for the circumstance.

We designed CS2.0 with tools specifically tailored for creating an intelligent digital ecosystem where information can be shared easily and intelligently and where improvements can be made by anyone in your organization. You can find other tools that more or less equip you for each of these bases you'll need covered, but to build the code-free, co-creation-embracing ecosystem we're describing, you need them all, and they need to function together. The considerations outlined below are best handled using specialized tools for maximum efficiency.

When people across your organization are automating tasks and creating their own skills and microservices, your shared library continues to grow as a resource for other creators and IDWs in the ecosystem. This creates the conditions for your ecosystem to expand and evolve organically. It's not unlike a universe unto itself, with an ever-expanding number of microservices that can be sequenced to meet any problem.

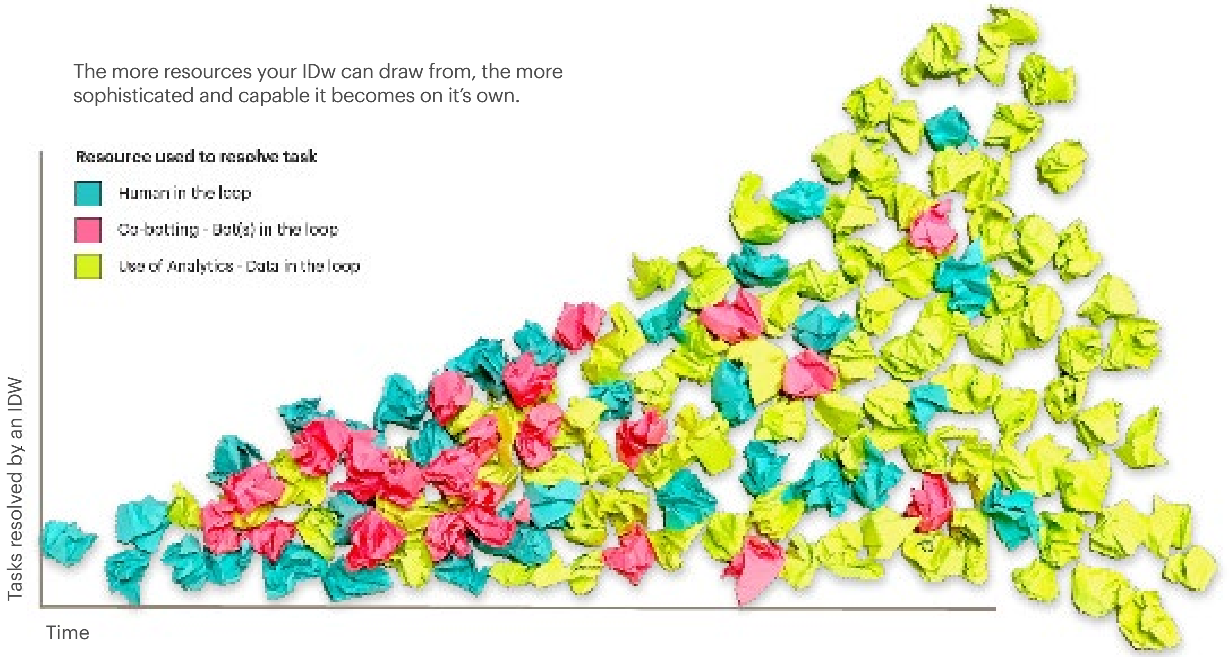
Here are some of the specialized tools you need in order to get to that state.

Humans in the Loop

Humans need to be able to monitor IDWs as the work, entering into the experience when the bot doesn't know what to do or needs guidance. The IDW can learn from the ongoing process of human-led refinement of automated tasks.

The continued evolution and expansion of an ecosystem relies on the human-in-the-loop process. Humans are a critical part of the ecosystem, working seamlessly alongside their IDWs, asking one another for help, querying, and establishing recommended responses and actions.

The more resources your IDw can draw from, the more sophisticated and capable it becomes on it's own.



In order to contribute to AI-training and step in where human-touch is needed, people need the ability to moderate IDW-managed conversations in real-time.

From these in-experience interventions, IDWs can learn from the live human-to-human interactions—as the member of your organization guides a user to the next step, the IDW gains new contextual data. The knowledge and skills retained by IDWs through in-line training can be leveraged across your organization.

The more time IDWs spend learning from their human counterparts, the faster the ecosystem can evolve.

They becomes more capable and requires less human intervention, freeing humans to move on to automating more tasks. Opportunities also emerge for something we call co-botting, where people and IDWs design or modify skills together. This could be as simple as a human realizing they should train an IDW on how to collect payment before a service is rendered.

On the flip side of that example, an IDW looking at analytics data and seeing that a significant number of users are requesting the option to prepay can reach out to a human peer for training on how to complete the transaction.

OneReach.ai

Shared Library

As we've pointed out multiple times already, the shared library is the central resource of your ecosystem. Everyone draws microservices, skills, and flows from it, and when these elements are customized for new services, that information becomes part of the shared resource. The shared library is the most effective way to move into hyper-automation, supplying your organization with best practices to scale knowledge sharing, accelerate development, and, at the same time, take control of security, compliance, monitoring, best practices, consistency, and scalability.

Actionable Documentation

It’s crucial to have accurate documentation for your shared library. Documentation enables members of your organization to understand which skills and microservices are available and operational as well as how to use them as they create new solutions.

Quickly Build and Iterate on Primary Skills

As you’re creating IDWs, you need to be able to quickly build and iterate on primary skills, like being able to operate over any number of specific communication channels (Slack, phone, SMS, Alexa, email, web chat, etc). Understanding natural language is also a base skill for an IDW, as is being able to include a human-in-the-loop when necessary.

Views, Dashboards, and Widgets

It’s imperative that you have the ability to generate custom reporting views, dashboards, and widgets that tie into your conversational experiences. These are used to trigger automated tasks and can offer real-time analysis and adaptations driven by your data. Equip your conversational applications to adjust the experiences they offer as they are happening, based on analysis and triggered events in customized reports.

Draw from Multiple Knowledge Bases

If you’re attempting to integrate an existing conversational application into your emergent ecosystem, you’ll need a tool that can give IDWs access to multiple knowledge bases—a knowledge garden, if you will. For example, imagine an IDW that is equipped to utilize a knowledge base your HR team built, as well as a knowledge base run through IBM Watson and one that you’re licensing from your HR Applicant Tracking Software. At some point, you might also need to bring a third-party application into the mix so, once again, you need a tool with the flexibility to do so. CS2.0 was designed with this capability in mind, making it easy to not only integrate third-party applications but to also build your own internal knowledge bases.

Easy Access to Stored Data

It’s important to have easy, permanent, accessible storage for the data your conversational applications captures and uses with low latency. You want tools that are designed and optimized to make hosting and accessing data easier on the back end, as users interact with your IDWs. By connecting your automated experiences to APIs or your own files, sheets, and data, you can support their flows. You’ll need to be able to push, pull, and store data without writing a line of code.

Own Your Roadmap

If, halfway up your Everest, you realize there’s a crucial tool you’ll need to get to the next checkpoint, it’s far better to have the ability to forge the tool right then and there. Radioing back to basecamp and asking someone to send it up to you means valuable time squandered. CS 2.0 allows you to build tools as you need them, rather than relying on a platform’s development team and losing the time it takes for them to generate a solution.

Use Rules Based AI and Neural Networking Together

In the realm of AI, there are often two proposed pathways or models: rules-based or neural networking. There is complexity and ongoing discourse surrounding each approach. Suffice to say that with the right tact and tools, you can combine elements of each, allowing your experience designers to create solutions far faster by reducing the need for massive amounts of training data.

Loading historical data into a machine learning model can create a generalized version of the conversations you’ve had, but doesn’t necessarily create good experiences. With the learning capabilities of neural networks and the power of rule-based AI, XDs can adapt to new settings and problems with far less data. Add human-in-the-loop systems to the mix, and you can cut up-front training data needs to almost zero.