Gartner

Architecture of Conversational AI Platforms

Magnus Revang, VP Analyst
Anthony Mullen, Senior Director Analyst
Van Baker, VP Analyst

Architecture of Conversational AI Platforms

Published 14 July 2020 - ID G00723272 - 35 min read

By Analysts Magnus Revang, Anthony Mullen, Van Baker

Initiatives: Artificial Intelligence **and 2 more**

Conversational interfaces are changing how we relate to machines, and application leaders need a strong understanding of this paradigm to stay ahead. This note looks at conversational AI platforms for chatbots and virtual assistants, through the lens of a common conversational architecture.

Additional Perspectives

- Summary Translation: Architecture of Conversational AI Platforms (11 August 2020)

Overview

Key Findings

- The conversational platform market landscape is hard to navigate, due to a growing number of vendors with a wide variety of applications, combined with inflated customer expectations and vendors' tendency to overstate their capabilities.
- Most end-user clients assume that artificial intelligence (AI) capabilities play a larger part in the overall platform, which creates gaps in expectations.
- Many offerings lack the features needed for more sophisticated implementations. Closing the gap will require substantial effort that is unlikely to be possible for niche vendors without sufficient staffing or funding.

Recommendations

Application leaders looking at how AI conversational platforms are evolving should:

- Choose vendors tactically through 2021 and 2022, as the rapid pace of maturation in conversational technologies precludes making strategic choices.
- Use the logical architecture map to understand the capabilities of conversational platforms, and align vendor offerings with their needs to avoid being stuck with a platform that doesn't fit requirements.
- Choose a platform approach to conversational capabilities, because the underlying architecture supports and enables a wide variety of use cases.

- Select only solutions or capabilities that have a robust learning loop with dialogue management as part of the architecture, enabling continuous improvement.

Analysis

In 2017 and 2018, there was a rapid increase in the availability of conversational platforms. The market is still crowded, consisting of large internet companies, enterprise software vendors, startups, and traditional call center and customer service software vendors (see "Market Guide for Conversational Platforms"). We estimate that there are more than 1,500 conversational platform vendors worldwide, and, although we are entering a period of consolidation, the vendor landscape will continue to be too large and volatile for strategic choice until 2022. Due to the constraint of language support, dramatic consolidation might never happen.

Enterprises are looking to solve a variety of use cases using conversational platforms. Regardless of the vendor targeting a role as a horizontal vendor, specialist vendor, an eco-system component or middleware — the underlying logical architecture remains the same. Customers looking at multiple use cases in the enterprise will benefit from a platform approach, so synergies between capabilities and skills.

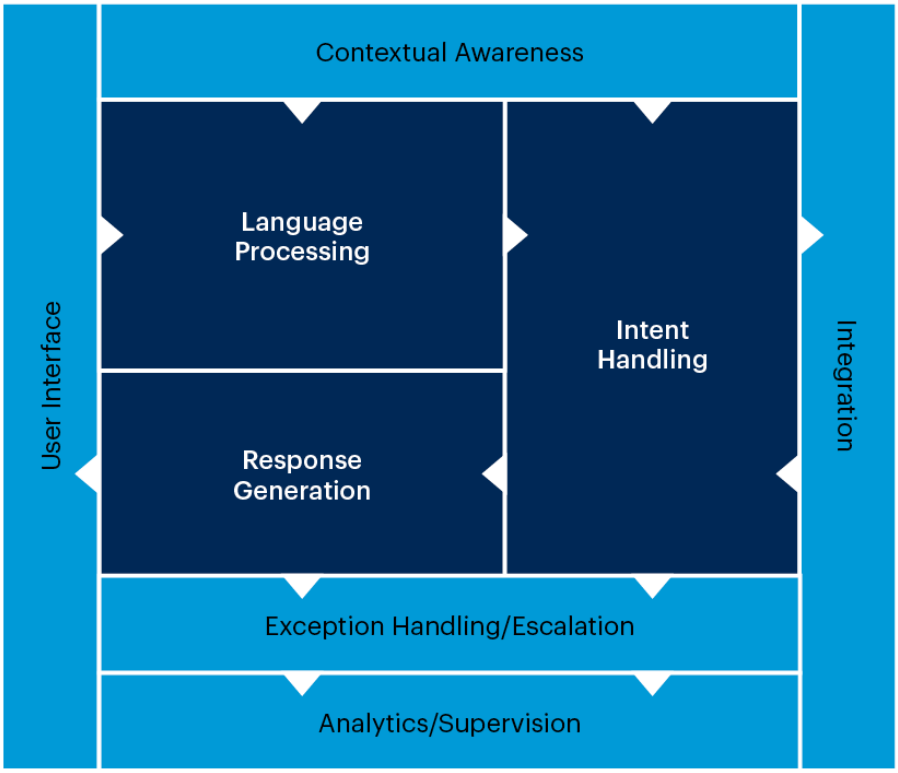
The simplest architectures and capabilities are relatively easy to replicate. More sophisticated architectures require much more customization and custom development. This creates a huge gap in the market between what different vendors are capable of. There is a correlation between the required sophistication of architecture and capabilities and the target sophistication of the implementation.

As we break down the capabilities, we are noting those present in almost all offerings, as well as those present in only a few. We are also noting capabilities that are being researched and may not even be in the market currently, but that we believe will be present in the future.

Figure 1 presents a representation of high-level architecture.

Figure 1. High-Level Architecture

High-Level Architecture



Source: Gartner
723272_C

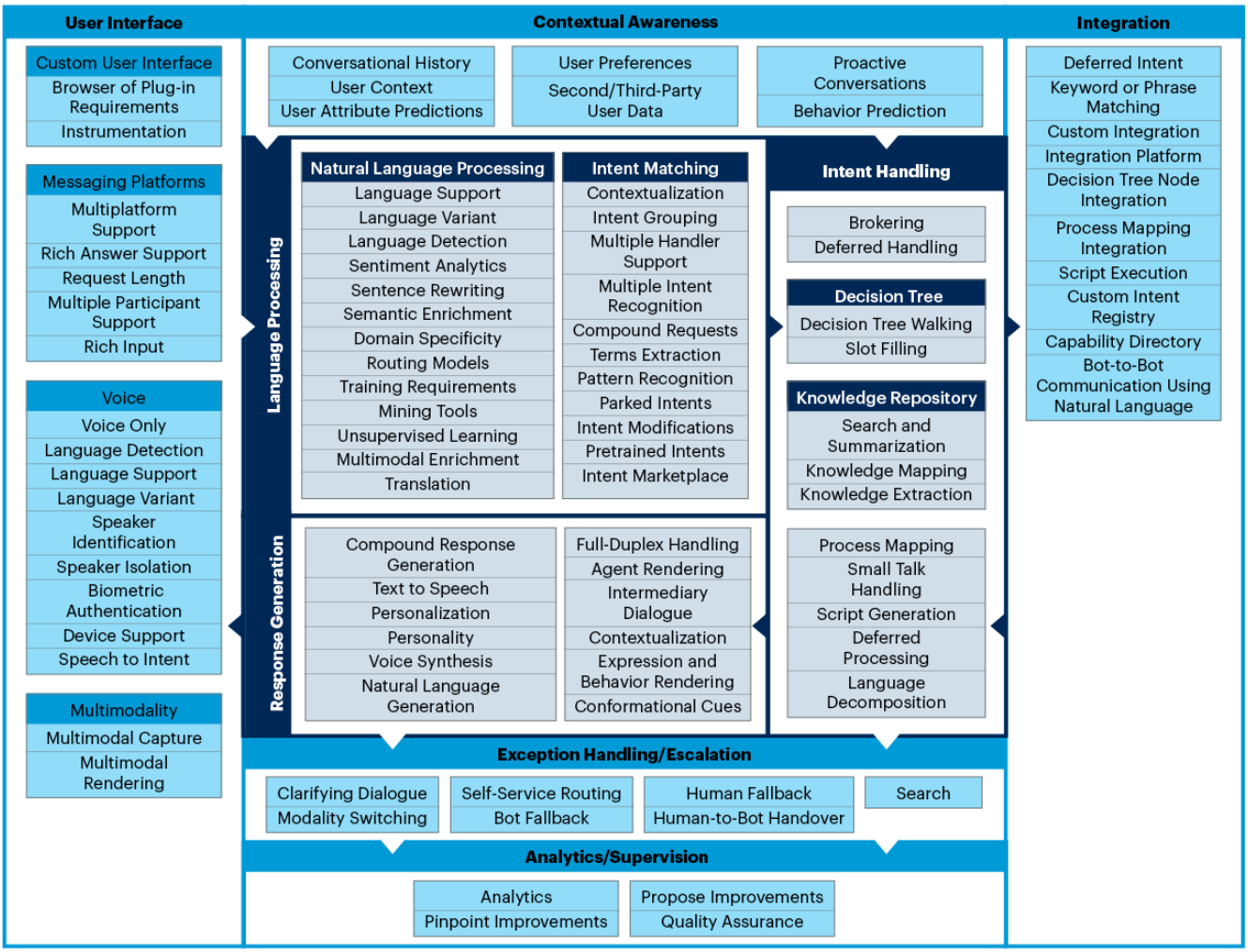
The high-level architecture of all types of platform is identical. A user interface (UI) is used to capture either voice or chat input, which is processed and passed along to be handled. Next, a response is generated (in many cases, just passed along from handling) to the UI.

The simple offerings we see in the conversational platform space consist of a channel integration connector, a natural language processing (NLP) engine, mapping of intent and a decision tree that gives responses. However, vendors that offer only this will not keep pace with the future requirements of conversational platforms.

When we expand the architecture to cover all capabilities we've cataloged so far, we see greater complexity and sophistication (see Figure 2).

Figure 2. Expanded Logical High-Level Architecture (Capabilities and Components)

Expanded Logical High-Level Architecture (Capabilities and Components)



Source: Gartner

723272_C

Understanding the high-level architecture and its capabilities is essential to enable users to evaluate and compare vendors. Because the market is evolving quickly and remains volatile, this becomes especially important.

Capabilities

User Interface

Several capabilities contained in the UI depend heavily on modality, with different considerations needed for chat- than for voice-enabled interfaces. In many chatbot and virtual assistant implementations the UI is not necessarily part of the platform. The conversational platform simply acts as a user on the communication and messaging platforms that human users already use to communicate among themselves (Facebook Messenger, Kik, WeChat, WhatsApp, Slack and Telegraph, as well as other messaging platforms, including email and website chat).

Chat

Chat is an interface that captures typed dialogue between two or more participants. All conversational platforms support one or more types of chat. Some leverage other chat platforms, some provide their own and some do both.

Custom UI

- **Browser or plug-in requirements.** Chat implementations sometimes use browser capabilities, such as WebRTC, which has varying support in browsers, or browser plug-ins that may not be supported or may create significant barriers for users. Evaluate the support for web browsers.
- **Instrumentation.** Whether web-based chat or a module that can be plugged into mobile apps, the ease of instrumentation is an important factor to consider. Although some solutions might only require a single JavaScript tag, others might need deeper and more time-consuming instrumentation.

Many other considerations for messaging platform integration apply. Think of the vendor's own chat interface as a separate messaging platform for the purposes of evaluation. Some vendors controlling their own chat implementations claim they have a conversational interface, yet all the user is doing after the initial request is clicking on predefined question options. Although this may suffice for certain use cases, it is not scalable to others. What more, such implementations are merely click-bots and do not constitute a conversational AI platform.

Integrations With Messaging Platforms

The rise of messaging platforms, such as WhatsApp, Messenger, WeChat, Teams and Slack, coincides with the rising hype surrounding conversational platforms, especially chatbots. However, messaging platforms also include older technology, such as email and SMS, as potential channels that a conversational platform can use.

- **Multiplatform support.** What messaging platforms does the conversational platform support? Although some vendors claim to support multiple platforms, you may need to develop or tailor a chatbot for each one. Different platforms might support different capabilities in input and output, a one size fits all can also become a solution only supporting a minimum set of capabilities (see rich input and rich answer support bullets below). Don't forget SMS and email as potential platforms as well. Email especially, might need a lot of special provisions, due especially to different lengths.
- **Rich input.** Chat usually supports only text; however, sending pictures or other nontextual information would be a natural evolution. Increasingly, messaging platforms are allowing this. Depending on the media, this can potentially involve a lot of other AI services and increase complexity exponentially.

- **Rich answer support.** Although all messaging platforms support simple, text-based responses, there is also varying support for richer responses. Examples include WeChat's ability to serve mini applications as a reply, or iOS Messenger's ability to serve limited interaction elements, such as graphics and buttons. Use of the unique capabilities of the messaging platform — which users might expect — needs to be balanced against the need to support multiple platforms.

- **Request length.** Consider the potential length of requests with chat messaging platforms, especially if you include email as a way to communicate with the conversational platform. If the platform's NLP engine is optimized for short requests and single intents, applying that platform to email, which lends itself to long descriptions and multiple intents, would be a bad match. That market offering has limited support for email. This is presumably because the complexity in handling the request increases exponentially as the length of the input increases.

- **Multiple participant support.** Messaging platforms support group chat. Make sure that this can be accurately detected and accounted for, because it might lead to novel use cases. A particularly interesting scenario for conversational platform use is to have a conversation between two humans, who, after a while, invite a chatbot into their conversation. The chatbot then can potentially have the whole conversation up to that point as context for further interactions.

Voice

Conversational platforms able to handle voice input offer varying degrees of capability. Over the last couple of years we've seen a move away from vendors having their own speech-to-text (STT) engines to more partnerships with existing vendors — or middleware capabilities of being able to integrate with multiple. Some conversational platforms differentiate themselves by being voice-first or even voice-only.

- **Voice only**. The ability to handle all interactions using voice without ever falling back on presenting rich output, such as search result lists, pictures and maps that require a screen.

- **Language detection.** The ability to detect what language is spoken and automatically switch to an engine supporting that language. In many cases, language has to be explicitly set either by the user or in the configuration of the platform.

- **Language support**. The ability to handle interactions in particular languages. Support for languages needs to be evaluated on quality, because variants, dialects, slang and accents are all capable of confusing the STT engine.

- **Language variant.** The ability to handle interactions in different variants or dialects of the same language — for example, French and Canadian French, Norwegian Bokmål and Norwegian Nynorsk, or formal and casual Japanese.

- **Speaker identification**. The ability to identify different speakers. This is especially important in multiple-user scenarios in which contextual data is being used for language processing, intent handling and/or response generation.

- **Speaker isolation**. The ability to not only identify, but filter on only one specific user from the background noise of additional voices.

- **Biometric authentication**. The ability to identify and authorize a user based on voice patterns. This is important if the conversational platform has requirements for security or privacy, or if risk for possible fraud needs to be reduced. More sophisticated solutions, with different levels of access for different users, can also be envisioned. They can exist both as an active component, where authentication is explicit — or as a passive component where the authentication happens in the background, possibly triggering the need for additional authentication mechanisms.

- **Voice pattern analysis and enrichment**. A lot of information and emotion is conveyed outside the words we use, through varying our tone, pace, pitch and other factors. By doing voice pattern analysis and enriching the generated text with additional meta information, some voice UIs are attempting to account for this additional information. In these cases, the fact that the platform is capable of doing this doesn't mean that it automatically accounts for this information in the generation of output. It merely adds the possibility if you're prepared to take on additional complexity.

- **Device support**. Voice recognition works well when all factors of the environment and hardware can be controlled. However, unknown variable quality can become part of the chain. This includes environmental noise, not having control of the microphone or sampling rate from a device, or attempting to do far-field voice control without the appropriate hardware. When this happens, the quality of voice recognition degrades. Knowing the scenarios of use and the degree of control over hardware is important to determine whether a voice interface can be a viable option.

- **Speech to intent**. Voice approaches that bypass an intermediate textual representation before determining intent is becoming slightly more common. Vendors that are voice-first or voice-only might offer this, and it offers the advantage of having a single level of uncertainty, instead of both uncertainty in the transcription and intent classification.

Multimodality

Conversational platforms will naturally focus primarily on dialogue in the form of chat and voice, so capturing other sensory data has the potential to improve the accuracy and the quality of the experience. While not common in today's platforms, we predict it will be a major differentiator over the next five years (see "Designing Conversational Experiences for Chatbots and Virtual Assistants").

- **Multimodal capture.** Voice recognition is likely to be augmented with data from video taken by a camera and other sensors. We are starting to see support in conversational platforms for gesture recognition, facial expression recognition, face recognition and biometric authentication. In the case of stand-alone devices, such as home speakers or in-car systems, additional specialist sensors might be built in to improve accuracy and the overall experience. Support for these additional signal sources, as well as for processing them needs to be mirrored in the architecture.

- **Multimodal rendering.** This is the capability to reply in chat or voice, as well as render other means that add to the exchange of information. This could be simple body language rendered on a virtual agent or expressions by a physical robotic assistant.

Processing

Behind the UI, a conversational platform needs to process input before it's able to generate the response. This part is required of any conversational platform. In many cases, it's also the only part that contains machine learning (ML) (see Note 1), even though the product is labeled as an AI product. Finally, the processing is, in many cases, done by a white-labeled — perhaps customized — engine or API from another vendor or multiple vendors.

Processing can be split into two steps: NLP and intent matching.

Natural Language Processing

The NLP step is where the input text, along with additional information from the UI and contextual awareness cues, is processed for the conversational platform to understand.

In the case of voice-enabled UIs, language and language variant support might be tightly integrated; however, in many instances, voice support simply converts the input from speech to text. NLP has to be performed on the textual output from the speech recognition.

Note: For the purposes of this research, NLP and intent matching are two steps; in reality, they are tightly integrated and difficult to separate in actual implementations.

Typical capabilities of NLP include:

- **Language support**. What languages are supported.

- **Language variant.** Different users have different styles when writing in chat interfaces. For some languages, different variants and even dialects may need to be supported — for example, American and British English, Norwegian Bokmål and Norwegian Nynorsk, or French and Canadian French.

- **Language detection.** Does the language need to be explicitly chosen by the user, or is the language automatically detected from the text? This is especially important when several languages overlap the same geography, such as in parts of Europe or Asia. In cases of language

variant detection, between different variants of English for example, this can be next to impossible.

- **Sentiment analytics.** Categorizes and identifies opinions expressed in the phrases the user is writing and attempts to derive the user's attitude toward topics, products or services. Sentiment analytics is often a separate service that works synergistically with the conversational AI platform (CAIP).
- **Sentence rewriting.** Parses phrases and modifies them before they are processed again for intent. This is a way to handle common challenges, such as misspellings, slang, synonyms or even sentence structures (e.g., double negatives). This can greatly increase accuracy in intent matching.
- **Semantic enrichment.** In the NLP step, text is typically enriched with semantics based on the internal knowledge base of terms and expressions (for example, tagging names, companies and actions mentioned in the text). NLP engines show a great variety of sophistication in this step, which varies a great deal, even among individually supported languages in the implementation.
- **Domain specificity.** Several layers of specialization are possible in the NLP engine. A typical, general-purpose vocabulary, out of the box, is likely to be ill-suited for most implementations. In most projects, a great deal of time is spent doing supplemental training to get the NLP engine to an acceptable level of performance. Domain specificity can take the form of:
  - Industry specificity — having a vocabulary tailored to understand banking, insurance or travel.
  - Purpose specificity — having vocabularies tailored to understand in the context of being used for IT service desk or calendar scheduling.
  - Customizable specificity — being able to manually configure synonyms, terms and phrases
  - Trained specificity — the vocabulary that's the final result of training the NLP engine with training data.
- **Routing models.** A possible variant on domain specificity is a broker pattern, in which an engine that specializes in simply detecting the domain will qualify the input to one or more underlying domain-specific solutions. See "Use Master Chatbot to Improve Conversational Experiences."
- **Training requirements**. Even if an implementation has good performance out of the box on a specific language, it might need a lot of additional training data to achieve the expected results. In some instances, that training data might not even exist. Thus, a customer might be stuck with a poor-performing implementation until enough training data can be gathered and processed.
- **Mining tools**. Some vendors have begun bundling mining tools that allow for mining datasources to enhance the language models.

- **Unsupervised learning.** Does the NLP engine have to be trained with training data only, or will it be able to adjust the model after deployment based on the result of the conversation with the user? In most cases, the unsupervised learning will still be an offline activity, but some chatbot implementations have tried "real-time" unsupervised learning. Be aware of potential "attacks" against a continuous, unsupervised-learning NLP, because it can be taught to respond wrongly with a coordinated effort. Proceed with caution with unsupervised learning approaches.

- **Multimodal enrichment.** This is taking data collected by other sensors to enrich the processing and results. For example, facial expressions may enrich a statement with information that makes it more likely to be interpreted a certain way, such as ironically.

- **Translation.** Will take a user's phrase in one language and translate it to a language that can be handled. This can be used in a variety of ways. The main issue to be aware of is that translation will never be as good as native understanding. Vendors trying to increase their language support by adding a translation step should be open about what they are doing, but that's not always the case. Gartner does not recommend using translation.

Intent Matching

Intent matching is where the processed input is matched to the appropriate handler of the request. This usually uses ML.

Note: For the purposes of this research, NLP and intent matching are two steps; in reality, they are tightly integrated and difficult to separate in actual implementations.

Several capabilities are possible:

- **Contextualization**. The ability to make contextual cues part of intent matching. Simple architectures will not consider context when matching.

- **Intent grouping**. Grouping of intents is important from the maintenance perspective of a large implementation, and for handling the scalability of a platform. Vendors are increasingly moving toward recognizing intent-groups first, then recognizing individual intents within that group as a secondary step. This is implemented either as hierarchies of intent matching models, or as networks of bots talking to bots. This can greatly help with scalability, especially in the case of deep-learning based NLP, as well as in the computational cost of retraining after changes or added intents. Grouping of intents enables whole groups to be enabled or disabled. This is a prerequisite capability for intent marketplaces.

- **Multiple handler support.** The simplest implementations have just one handler, which is usually a decision tree. The simplest intent matching consists of just matching input to the appropriate point in the decision tree. Additional complexity is introduced if it's possible to match to multiple types of handler.

- **Multiple intent recognition and prioritization.** The simplest implementations support just one intent per input, which means that requests with multiple intents need to be explicitly accounted for. If supporting longer-form formats, such as email, the ability to handle multiple intents becomes essential. Take this simple example:
  - "I want to order a pizza, but I'm unsure if you can deliver to my address?" The answer to this would be greatly improved if the intent matching is able to detect both "want to order" and "check delivery coverage." Being able to prioritize the two intents is important to pass along to the handler.

- **Compound requests.** Similar to multiple intents, compound requests are multiple; but instead of affecting each other, they are completely separate. For example:
  - "I want to order a pizza, and I also want a soft drink, as well as the movie on offer."

- **Terms extraction.** A user might have the same intent, but the request might contain additional information for the intent handler. For example:
  - "I'd like to order a pizza with marinated chicken, four cheeses and tomatoes." The intent is "order pizza," but a list of ingredients is also given to the handler.

- **Pattern recognition.** This is more advanced than terms extraction, in that the intent matching is capable of matching intent with unknown terms in it by looking for particular patterns in the request. An example: "Play a song by Prince on Spotify." Terms extraction would require knowledge of all potential artists; but when doing pattern recognition, the nature of the pattern would indicate the intent, rather than the specific terms used. More-advanced implementations of pattern recognition would take context into consideration as well.

- **Parked intents.** This is the ability to recognize when a user asks for something different in the middle of a dialogue. Users need to be able to park the current intent resolution, handle the new intent and, in the end, get back to the parked intent to continue. Advanced versions of this would be able to handle layers of this, where multiple intents can be parked and the ability to use the new information to influence resolution of the parked intent.

- Intent Modifications. The ability to recognize phrases that modify the original intent. For example:
  - "I'd like to order a pizza with beef, onion and pineapple." … dialogue continues for a while … "Know what, can you take the pineapple off, I don't want that topping." … dialogue continues to resolution …

- **Pretrained intents.** A library of intents and the ability to recognize them for a particular use case or domain. A pretrained intent library can greatly improve the time to market and reduce the effort involved. Quality, precision and size vary greatly among vendors.
- **Intent marketplace.** This is the availability of a marketplace in which intents can be shared and sometimes even monetized. A real intent marketplace enables you to pick multiple intent libraries, and to modify them to your needs. It differs from pretrained intents in that it is an ecosystem of more providers — and doesn't rely on a single vendor to create the intents.

Since intent matching is commonly based on ML, take special care when evaluating the cost of maintenance and governance. As a rule of thumb, the more advanced the intent matching capabilities used, the more training data is required to make it work at an acceptable performance level.

Contextual Awareness

Being aware of the context of use can greatly enhance a conversational platform's ability to successfully match intent.

- **Conversational history.** The ability to learn from previous conversations and reuse that information in future conversations becomes increasingly important as frequency of use increases (such as in virtual assistants and some chatbot use cases).
- **User context.** This is the ability to take into account external contextual cues. Examples include on what page a website chat is used, previous pages visited or geographical location gauged from the mobile phone GPS.
- **User preferences.** These are especially important for implementations that see frequent use. There might be a requirement for users to access preferences and turn off certain capabilities that they are not comfortable with or want enabled. This can take the form of settings screens in an app, or even the ability to set preferences through the dialogue (e.g., "I prefer that you call me Bob").
- **Second/third-party user data.** This is the ability to leverage outside data about the user. For example, it can take the form of CRM data or information from a public profile on Facebook.
- **Behavior prediction.** This is the ability to predict the behavior of a user based on past interactions or past interactions with others. This might allow the platform to skip unnecessary steps in dialogue, or contribute to upsell or problem solving.
- **Proactive conversations.** One exciting development is the ability of conversational platforms to initiate conversations, instead of only responding to users. To be able to do this, the platform would need integrations with some kind of event processing and either explicit or implicit rules for triggering such a conversation.

- However, the line between helpful and irritating is hard to navigate. This kind of functionality requires a careful consideration of the negative consequences, as well as extensive testing and validation. Customer journey mapping is a common way to ensure quality of proactive conversation approaches.

- **User attribute predictions**. This attempts to classify the user, based on writing patterns and use of words (and even speed of reply) into attribute groups. Common use cases in marketing and sales predict the demographic group, to which the user belongs, to give relevant upsale offers as part of the conversation.

Handling

Using decision trees to handle requests is, by far, the most common handling method in conversational platforms. This often comes as a surprise to customers, because vendors like to attach the labels “machine learning” or “AI” to their offerings. In the case of a decision tree, there is no ML involved, except in how requests are mapped from natural language to the intent that signals the entry point into the scripted dialogue.

This is a field where we see a lot of potential for innovation. We fully expect to see new ways of handling in the future. Other capabilities, such as multiple and compound intents, will quickly grow decision trees to unmanageable sizes, necessitating the need for new handling methods.

- **Brokering.** In cases in which a platform has multiple ways to handle intents, a mechanism to broker intents among the different handlers is necessary. See “Use Master Chatbots to Improve Conversational Experiences.”

- **Deferred handling**. The easiest handling is to just pass the processed request along to a custom-developed service, like skill based implementations. For specific-purpose chatbots, this might be all that is needed, but other kinds of handling are required for most implementations. Even if a conversational platform has sophisticated handling capabilities, the option to defer handling of certain requests can still be a way to enhance the experience and enable truly differentiating experiences.

- **Decision tree walking**. By far the most common way of handling requests, this consists of a dialogue tree in which each node is matched to an intent, and contains potential subnodes to keep the conversation going. For each node, there might be an option to respond with a standardized response or pass along to an API, consider:

- How the tree is created and maintained (visual tools, in code, configuration files and even needing professional services are all possible)

- How sophisticated the tree can be

- How scalable the tree will be if it grows large
- How sophisticated the handling on each node can be, including the response

The main advantages are that decision trees are fully transparent, and answers can be fully controlled, thus it's comparatively easy to reflect consistent brand values and ensure regulatory compliance.

- **Slot filling**. A way to simplify decision tree design for transactional conversations, but it does require NLP/natural language understanding (NLU) capabilities to extract entities that will fill the slots. It works by specifying an end state (we need these seven pieces of information to run the transaction), and letting the platform automatically generate the dialogue necessary. It will then skip any information it already has from context, and handle the user entering three pieces of information in one phrase. Some platforms handle this in the engine itself, while others let you do slot filling in the dialogue design tool, and it gets converted to a regular hierarchical decision tree in the background.
- **Search and summarization**. Takes the phrase written by the user and turns it into a search query to run against one or more knowledge repositories. The least sophisticated solutions will typically just strip out anything but paragraph headers. In other solutions, the results from search are then sent to a summarizer and are presented back to the user.
- **Knowledge mapping.** The ability to have the answers outside the conversational platform and manually map the intents to the right place to find the answer. This can save time and effort on maintenance — for example, when the canonical answer is on the website.
- **Knowledge extraction.** Also called fact extraction, this is the ability to turn a request into a query. The relevant information is extracted out of a large knowledge or content repository (or multiple), and presented back to the user as an answer. Similar to knowledge mapping, except there is no manual work to map between intents and where the information is. Knowledge extraction can both be a preprocess — where knowledge is ingested and turned into question and answer pairs for internal representation, or it can be done at intervals and even runtime, depending on implementation.
- **Process mapping.** This is the ability to map conversations into steps in business processes, focusing the conversational elements on what is "current state" and what information or action is needed to move to the next step in the process.
- **Small talk handling.** The ability to gracefully deal with small talk attempts by the user, like talking about the weather or common greetings, such as, "How are you doing today?"
- **Script generation.** Also called query generation, this is the ability to translate output from the processing step into a more-formalized scripting language that is executed in a script engine. The script engine could enable other handling mechanisms and direct integrations. This is most

commonly used in augmented analytics solutions that offer conversational interfaces against business intelligence data.

- **Deferred processing.** Instead of just deferring the handling, the whole unprocessed request is passed along to another conversational platform or bot implementation on the same platform. Together with language decomposition, this can enable the handling of complex, general purpose requests by multiple, specific-purpose implementations.
- **Language decomposition**. This is the ability to decompose complex requests into simpler separate statements, which are then run as separate requests to different handlers. This would require the ability to compound answers in the response generation. For each type of handling supported, the amount of work involved and the toolsets to do that work should be evaluated.

Vendors that offer only deferred handling and/or decision tree walking will struggle to keep pace in this area. The research and development effort required to implement the other handling methods is exponentially larger, and many will not be able to cross that chasm. This favors vendors with a sophisticated approach to intent handling and dialogue management.

Integration

Although stand-alone chatbot implementations have a purpose, many use cases require integration with existing systems. There are several ways that this integration may happen.

- **Deferred intent.** After processing and matching intent, the request is passed along to a system that has registered itself to handle that particular intent. It requires the implementation to control the conversation, but allows for interchangeable services to execute the requests. An example would be having Uber, Lyft or the local taxi service register for a “get a car” intent. It requires the registering service to implement a known API to handle the deferred request.
- **Keyword or phrase matching**. Integration through registering a keyword or phrase that, when employed a user, triggers deferring of the handling to the service. An example would be, “Tell Spotify to play some Christmas music.”
- **Custom integration.** Custom integration simply means that integration needs to be custom coded for the implementation.
- **Integration platform.** In more-sophisticated platforms, a third-party or custom integration platform may be enabled to ease the consumption of APIs and the managing of integrations. If there’s a need to integrate many back-end systems, this might be a necessary capability.
- **Decision tree node integration**. The ability to specify a RESTful method call to be executed from a particular node in a decision tree. This allows simple integration from a SaaS-based conversational platform to available RESTful APIs.

- **Process integration**. Integration from an internal representation of the business processes in a process-mapping handler to a business process management (BPM) workflow engine that handles the workflow and integrations.
- **Script execution.** In the case of script generation handling, any services that require integration would need to implement an ability to execute the script. This allows for deeper integrations than APIs, but also requires more implementation work on the part of the service provider.
- **Custom intent registry.** This is the ability to train your service in recognizing a particular intent (which can be unknown to the conversational platform), and register that intent, along with the training to recognize it with the conversational platform.
- **Capability directory.** This is maintaining a directory of services that implement a particular API, one or more of which can be used when the capability is needed.
- **Bot-to-bot communication using natural language**. The ability of the conversational platform to turn requests into new requests that are then used to communicate with other bots (see "Maverick* Research: Machines Will Talk to Each Other in English").

Response Generation

Anything more sophisticated than prescripted responses needs capabilities for doing response generation — at a minimum, natural language generation (NLG).

- **Compound response generation.** In cases in which a platform supports multiple intents or compound requests and is able to handle them separately, there is a need for taking several answers and turning them into one compound response.
- **Text to speech**. In the case of voice interfaces, this is the actual audible voice generated by the platform. Most vendors use external services to do this.
- **Personalization**. Beyond basic personality is the ability to tailor the personality to the current implementations. In addition, there is the ability to take into account the writing or speaking style of the user, cultural cues and other factors, to personalize the response to an individual user. Almost no vendors offer this.
- **Personality**. The ability to add personality characteristics to automatically generated responses. Personality is of vital importance to the design of conversational experiences, and accurate projection of brand values can make experiences better for users.
- **Voice synthesis.** Ability to generate humanlike voice, advanced functionality would be multiple voices, variances of incantation based on context and support for multiple languages.
- **Full-duplex handling**. The ability to understand what the user is saying before the user is finished is important for voice. It allows for corrective statements, clarifying questions,

interruptions and confirmational cues in a fashion that's more natural for users.

- **NLG.** The ability to generate natural language responses, based on structured data or other inputs. NLG is important if there are handling methods other than decision trees. Most NLG is not very sophisticated, being semantic and template based. Neural networks may however see us in low risk use cases. Stand-alone NLG tools are most often used to respond to emails and other long form content, and not short form dialogue of chatbots. Assess the amount of training needed to have the NLG produce good responses, and the quality of the output from a readability perspective.

- **Agent rendering**. This is the generation of an agent, in the form of a humanlike body or face, or even a robot illustration. The generation includes facial expressions, lip syncing to voice, body postures and gestures. Consider this if there's a need for agents in the first place. Many vendors support this feature primarily as a way to brand their products.

- **Intermediary dialogue**. Sometimes processing, handling, getting data from integrations and generating an answer takes time. If cloud processing of voice is necessary as well, responses may be delayed by seconds, creating awkward gaps of silence in the dialogue that degrade the experience. Intermediary dialogue ("Hold on while I look that up") is similar to how a human would respond, and may help mitigate latency problems.

- We expect this to evolve further, with some conversational platforms starting a response without knowing the actual result until later in that response ("Looking at your accounts … [lookup complete at this point], your balance is $X").

- **Contextualization.** The ability to use the contextual capabilities in the platform to tailor the response.

- **Expression and behavior rendering.** The ability for agents to add expressions based on context, such as idle movements, waiting, anticipation, nodding and other feedback cues.

- **Confirmational cues**. The ability to ("aha," "hm," "huh," etc.) to express understanding, deal with confusion or guide the user through sounds. This is important in voice to achieve higher precision and more natural conversations.

Exception Handling

Exception handling, also called escalation, is the ability to route a request that is not understood, or poorly understood, to an alternative handling method. Considerations must be taken for what triggers an escalation. Is it simply when an implementation is unable to answer, or is it when the predicted Net Promoter Score 2 from the interaction falls below a certain threshold, requiring a human takeover of the dialogue?

- **Clarifying dialogue.** When faced with multiple possible intents or an intent below the confidence threshold, this is the ability to ask questions that will improve the confidence threshold. This is a notoriously hard problem to solve, and may involve a great deal of manual work to get correct.
- **Modality switching**. Instead of answering in the conversation, the ability to direct the user to an appropriate service, app or website where the request can be fulfilled.
- **Search.** The capability of passing along the request to a search engine that will present the user with search results. This will take the user out of the conversational paradigm, forcing a modality switch (below).
- **Self-service routing.** Routes to the appropriate self-service system, like a ticket system.
- **Bot fallback.** This is the ability to fall back to another bot. Typical uses of this is mixing multiple bots in the same system. Like a front bot doing broad intents, deep conversations and transactions — and a bot behind it that can do simple questions and answers.
- **Human fallback.** The ability to pass along requests to a human, who then takes over the conversation. More-sophisticated solutions not only pass to humans, but enable humans to pass back to the machine when the request has been clarified. Or present the human with several alternatives for the agent to select.
- **Human to bot handover.** When a human fallback is done, this is the ability for the Agent to hand back to the bot. This is useful in cases of transactional checkouts where agents should not handle things like credit card information.

Analytics/Supervision

All solutions should have analytics. More-sophisticated platforms also give you the tools to turn analytics into action and help you improve.

- **Analytics.** The ability to generate reports and look at the performance of the implementation. Considerations should be taken according to how the metrics are being used: Is the need just for reporting at regular intervals, or via real-time monitoring?
- **Supervised learning loop.** An interface to easily map missed intents to what the engine should have originally responded.
- **Pinpoint improvements.** The ability to pinpoint potential areas for improvement — typically similar requests that are not being handled and similar answers given by human employees on fallback. Although it pinpoints possible places to improve, it does not tell you what improvements to make.
- **Propose improvements.** The ability to monitor and propose new additions to the decision trees or other handlers. It often involves ML to give proposals and human supervision to approve

them.

- **Quality assurance.** The ability to ensure consistent quality, as the implementation scales. This includes monitoring the quality of intent matching, so training phrases that would make performance deteriorate would be flagged.

Even when doing a simple proof of concept (POC), analytics is of vital importance to learn and improve an implementation. Solutions that don't offer analytics are effectively running blind and should not be considered for any purpose.

Using This Research Effectively

There are a number of ways to use this research effectively:

- **Talking to vendors.** Use this research to establish a common taxonomy and vocabulary with the vendor; to cut through marketing and hype.
- **Scoping.** Scope the kinds of capabilities you are looking for; to find a service or product that's a good fit.
- **Evaluation.** Evaluate current capabilities, implementation approaches and the roadmaps of vendors to determine future viability — will they scale with your future needs?
- **Integration.** Determine the future integration needs of existing systems and how they might fit in a conversational platform.
- **General knowledge.** A tremendous amount of discovery is embedded in this logical architecture. It allows you to have conversations on an elevated level of sophistication with vendors, developers and designers.

Acronym Key and Glossary Terms

AI	Artificial intelligence
CAIP	Conversational AI platform
ML	Machine learning
NLG	Natural language generation
NLP	Natural language processing
NLU	Natural language understanding

SaaS	Software as a service
STT	Speech to text
UI	User interface

Evidence

1. Major companies, especially in the VPA market category, have done several acquisitions of startup companies that were working on gesture and emotion detection from regular webcam footage.

2. Net Promoter Score is explained in "What Is Net Promoter?" NICE Satmetrix

Note 1: Machine Learning and Curated Learning

In this context, curated learning is a manual and human operation in which the answers that the chatbot gives are written and maintained by humans, using an administration tool. There might be rules and development of scripts and/or integration with back-end systems as well.

In this context, machine learning is an automated approach to generating answers. For example, this can be observing human operators giving answers and using those observations to improve the chatbot's ability to automatically answer queries.

Recommended by the Authors

Guidance Framework for Evaluating Conversational AI Platforms

Use Master Chatbots to Improve Conversational Experiences

Designing Conversational Experiences for Chatbots and Virtual Assistants

Market Guide for Conversational Platforms

Market Guide for Natural Language Generation Platforms

Market Guide for Speech-to-Text Solutions

Using Conversational AI Middleware to Build Chatbots and Virtual Assistants

Market Guide for Virtual Customer Assistants

Maverick* Research: Machines Will Talk to Each Other in English

Recommended For You

Summary Translation: Architecture of Conversational AI Platforms

Cool Vendors in Conversational AI Platforms

Unlock AI Functions in Business Applications

5 Myths About Explainable AI

H2O, Critical Capabilities as of August 2020

Supporting Initiatives

Artificial Intelligence

Track

Application Architecture, Development, Integration and Platforms

Track

Application Leaders

Track

Learn more.
Dig deep.
Stay ahead.

Gartner's artificial intelligence insights can help leaders drive revenue growth and achieve business outcomes by improving quality, speed and functionality.

Learn more: gartner.com/en/information-technology/insights/artificial-intelligence

Become a Client

Get access to this level of insight all year long — plus contextualized support for your strategic priorities — by becoming a client.

gartner.com/en/become-a-client

U.S.: 1 800 213 4848

International: +44 (0) 3331 306 809

About Gartner

Gartner is the world's leading research and advisory company and a member of the S&P 500. We equip business leaders with indispensable insights, advice and tools to achieve their mission-critical priorities today and build the successful organizations of tomorrow.

Our unmatched combination of expert-led, practitioner-sourced and data-driven research steers clients toward the right decisions on the issues that matter most. We are a trusted advisor and an objective resource for more than 14,000 enterprises in more than 100 countries — across all major functions, in every industry and enterprise size.

To learn more about how we help decision makers fuel the future of business, visit gartner.com.

Gartner®