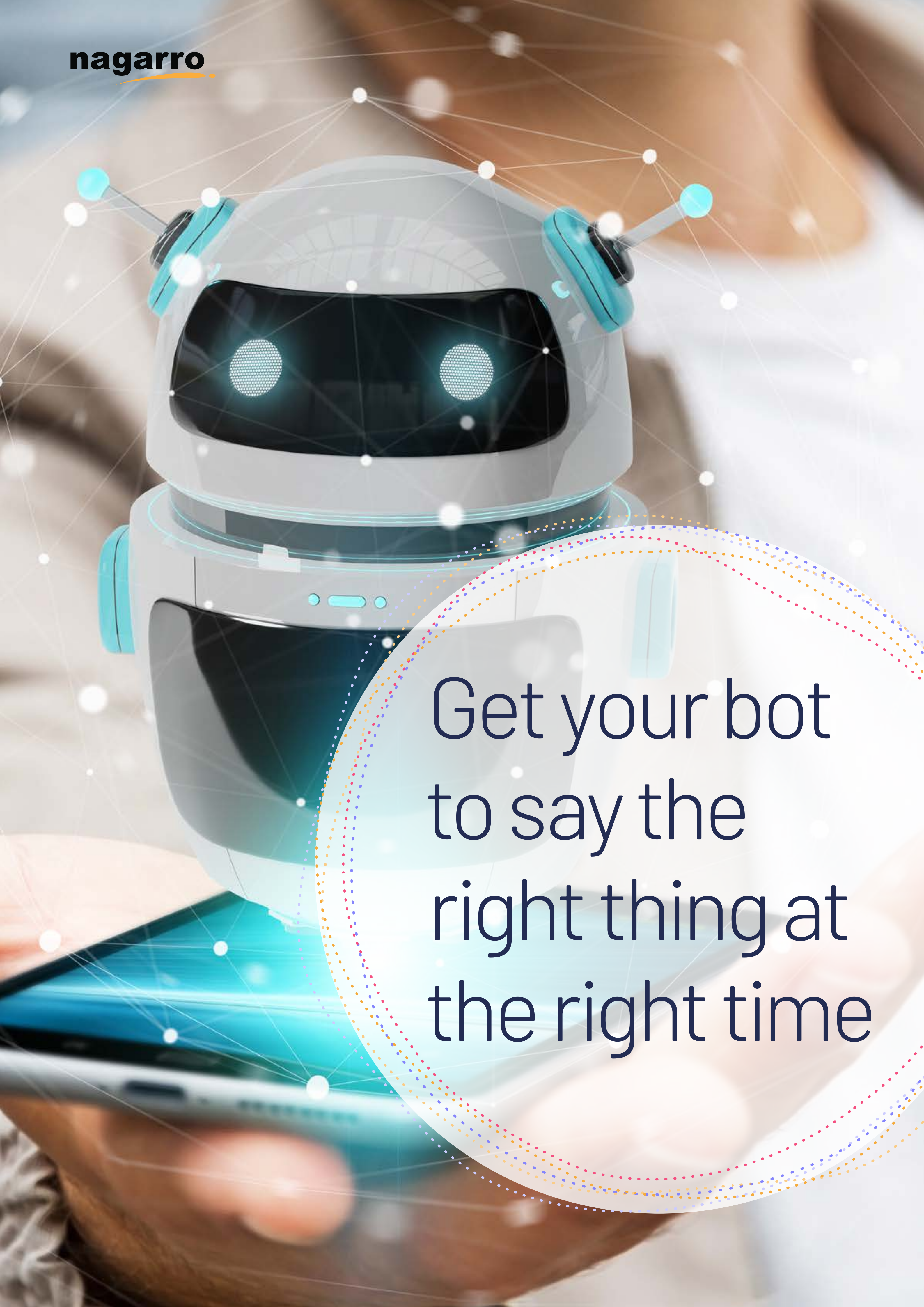
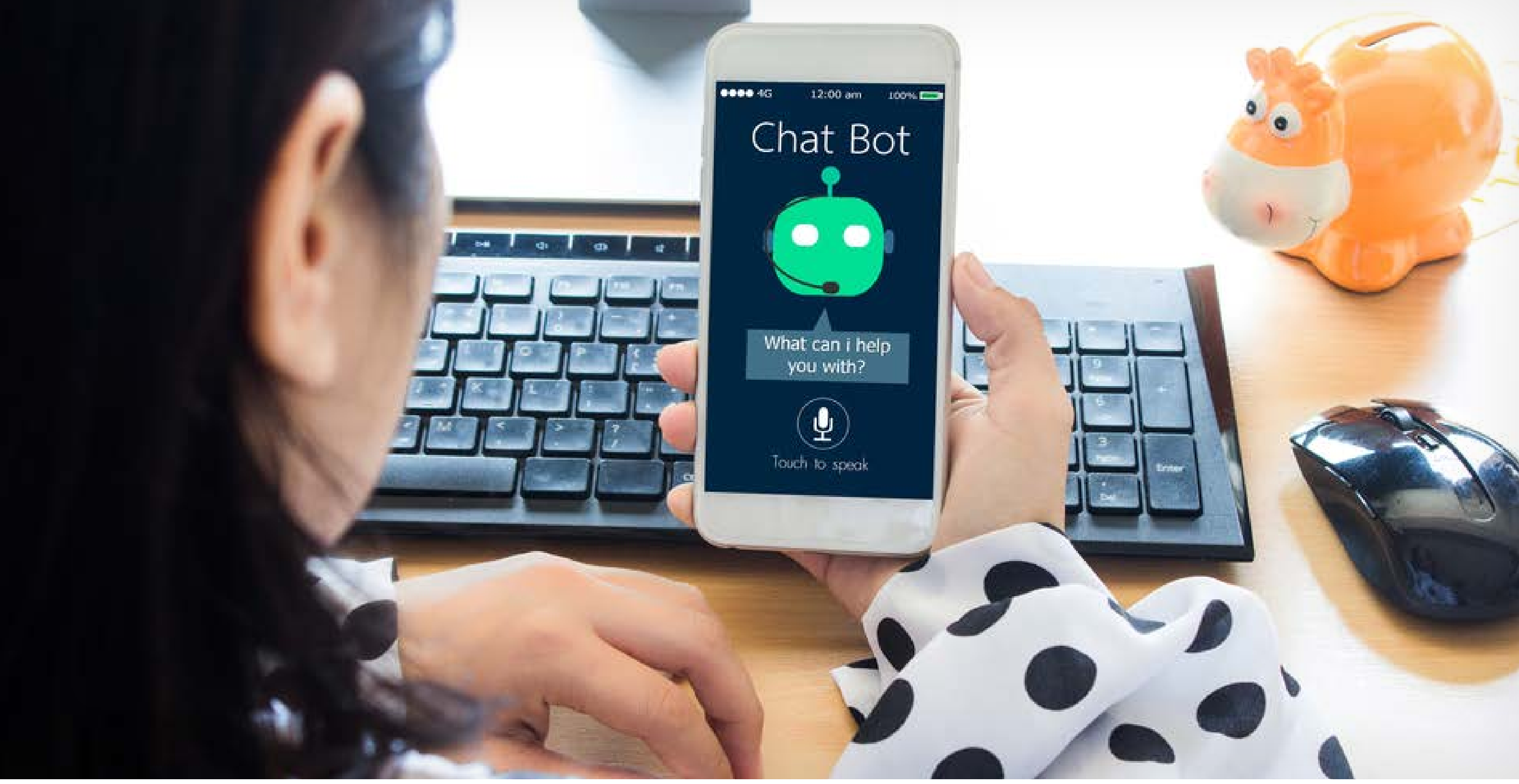
nagarro



Get your bot to say the right thing at the right time

Table of Contents



INTRODUCTION

Chatting with a bot, one learns the importance of real-time feedback and there is no limitation on user input, any user can type anything to a Chatbot. Ergo uncertainty has always been a key challenge for testers - be it with the ambiguous definition of requirements or with unstable test environments. But testing a chatbot adds a completely new level of uncertainty to a tester's life.

Numerous platforms and tools available for chatbot development, but no standardized chatbot testing strategy. The way testing is performed on chatbots differs a lot from the "traditional" testing (for example of an mobile application or web portal) due to the apparent randomness of a conversation with a Chatbot.

Testing numerous clients' chatbots and our own chatbot, we experienced that it is impossible to anticipate and cover all the situations that can happen during a conversation with a chatbot.

As we introduced learning components to chatbot (AI / machine learning, intent training), the chatbot evolved and changed its behavior compared to previous test runs.

Any user can type anything to a chatbot, so functionality, security, performance and exception handling need to be robust. This white paper aims to provide a thorough chatbot testing strategy taking quality experts beyond traditional approaches.

1. WHAT ARE CHATBOTS?

Techopedia defines a chatbot as "an artificial intelligence (AI) program that simulates interactive human conversation by using key pre-calculated user phrases and auditory or text-based signals." It is also known as a chat robot, talk bot, chatterbot, and an ACE (Artificial Conversational Entity).

In simpler terms, a chatbot is an assistant that communicates with us through text messages, as a virtual companion. It integrates into websites, applications or instant messengers and helps businesses provide a personalized experience to their consumers. Due to an ever-increasing demand, every customer wants an easy, 24x7 access to various online services, with instant and on-demand personalized (omnichannel) experience. Chatbots have emerged as a popular solution to these new requirements in today's world. Designed using AI and machine learning, chatbots simplify and improve the interaction between humans and computers.

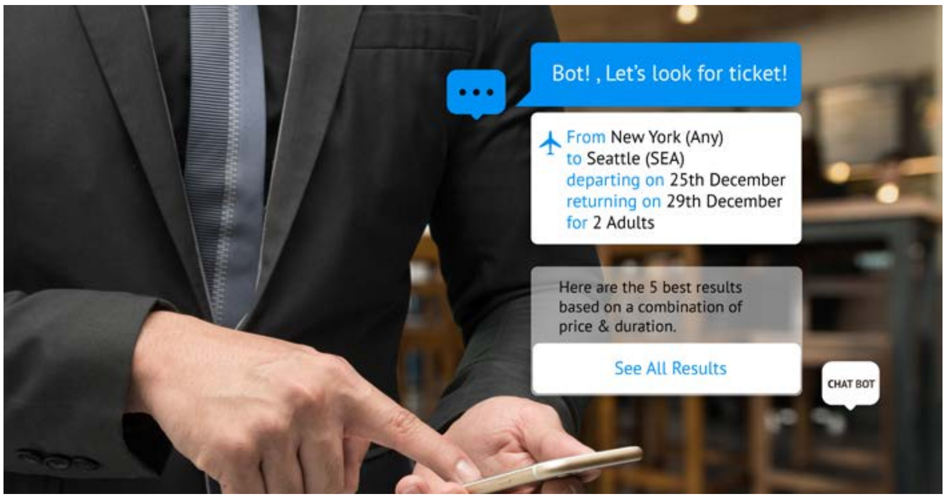
2. WHERE ARE CHATBOTS USED?

Chatbots are commonly used in situations which require simple interactions with a defined range of responses. They are extensively used for basic customer service and marketing systems that frequent social networking hubs and instant messaging (IM) clients. Chatbots are also deployed in operating systems as intelligent virtual assistants. For example, as Siri for Apple products and as Cortana for Windows. Dedicated chatbot appliances are also becoming increasingly common, such as Amazon's Alexa. These chatbots can perform a wide variety of functions, based on inputs provided by the users.

Chatbots provide the following benefits:

- Immediate response
- Simplification of complex activities
- Great appeal to non-tech-savvy audience
- Extreme personalization and contextual adaptability
- 24*7 availability
- Cross selling, affiliate marketing, lead generation, retail, etc.

According to an Oracle survey, 80% of businesses want either a rule-based or an AI-based chatbot by 2020. However, building a chatbot requires the same components across all platforms, technologies, and domains. The solution is to start the development with basic bot framework implementation in a project and then enhance the solution as per the required feature and business needs.


Bot! , Let's look for ticket!
From New York (Any) to Seattle (SEA) departing on 25th December returning on 29th December for 2 Adults
Here are the 5 best results based on a combination of price & duration.
See All Results
CHAT BOT

3. HOW DO CHATBOTS WORK?

Before we dig into the challenges of building a successful chatbot, let's first understand how a chatbot works. Let's understand the differentiators for various businesses and technologies. Broadly, there are three ways in which a chatbot can operate.

3.1. Matching patterns

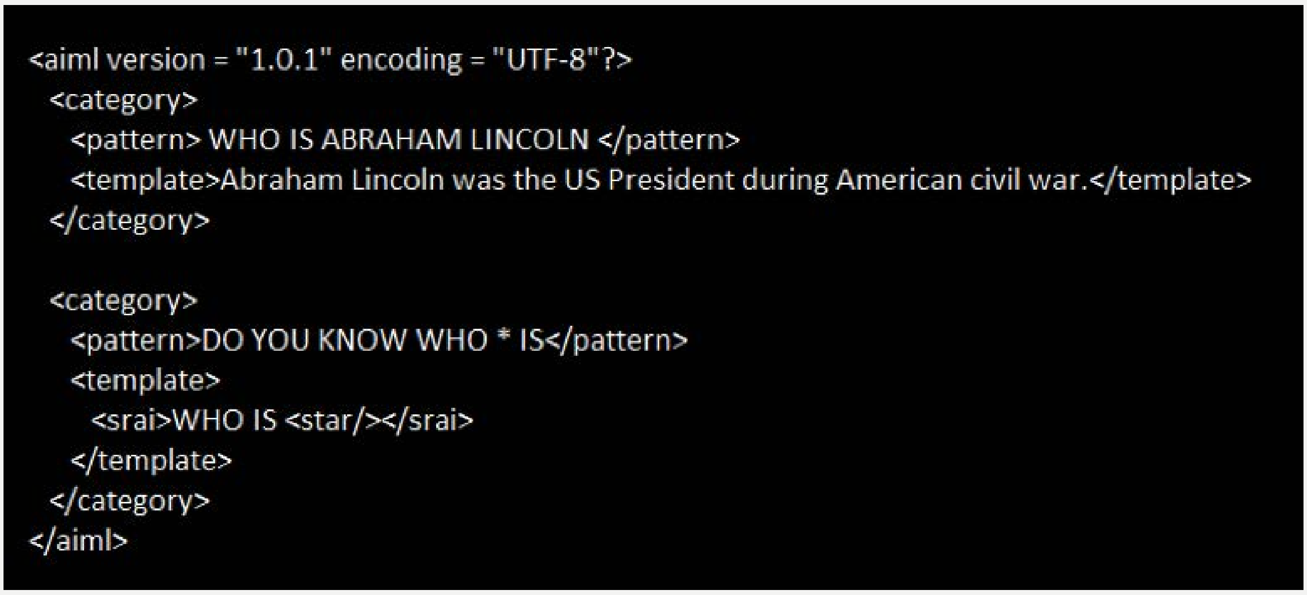
The earliest chatbots matched patterns to classify text and produce a suitable response for customers. The Artificial Intelligent Markup Language (AIML) comprises a defined structure of these patterns. Here is an example of a simple pattern matching approach:

The chatbot will know the answer because the given name is present in the associated patterns. Therefore, it will respond to anything pertaining to the same. However, the chatbot will not be able to go beyond the determined pattern. For an advanced level, the chatbot needs algorithms.

3.2. Algorithms

Chatbots need to have a unique pattern available in their database for each question posed to them. This pattern helps them to provide a suitable response for that question. Through what is known as the reductionist approach, algorithms reduce the classifiers and generate a more manageable structure. This simplifies the solution by reducing the problem. Let's assume a set of sentences is given and this set belongs to a class. With any new sentence that is entered, each word is counted for its occurrence and is accounted for its commonality. Each class is assigned a classification score. After compiling the scores of each class in this way, the class with the highest score is most likely to be associated with the input sentence. This is how a chatbot responds to queries.

With the help of equations, word matches are found by using some sample sentences for each class. The classification score identifies the class with the highest term matches, though it



```
<aiml version = "1.0.1" encoding = "UTF-8"?>
  <category>
    <pattern> WHO IS ABRAHAM LINCOLN </pattern>
    <template>Abraham Lincoln was the US President during American civil war.</template>
  </category>

  <category>
    <pattern>DO YOU KNOW WHO * IS</pattern>
    <template>
      <srai>WHO IS <star/></srai>
    </template>
  </category>
</aiml>
```

Figure 1: A simple pattern matching approach

For example, A sample training set

Class: greeting
"How you doing?"
"good morning"
"hi there"

Some sample inputs/ sentence classifications:

Input: "Hello good morning"
Term: "hello" (no matches)
Term: "good" (class: greeting)
Term: "morning" (class: greeting)
Classification: greeting (score=2)

also has some limitations. The classification score also signifies the most likely intent of the sentence but it does not guarantee that it is the perfect match. The highest score only provides the relativity base.

3.3. Artificial Neural Networks

Each sentence/input is broken further into different words and each word is then used as an input for the neural networks. The weighted connections are then calculated by different iterations through the training data. This activity is performed multiple times, with each iteration improving the weights to make it accurate.

Another user scenario: A user starts a chat by using different chat clients such as Skype, Facebook, Gmail, etc. on a web/desktop interface or through a mobile interface. This channel interacts with the bot connect services that would receive the message sent through the different platforms. The bot framework processes the business logic, which may require interaction with the backend system for information exchange, retrieval or to update and send responses to the user's queries. For a simple bot, it is sufficient to have the business rules which are created based on their requirements and interactions with the backend system. However, an intelligent bot, based on Artificial Intelligence and Machine Learning, also requires cognitive skills like NLP, utterance development, intelligent service, intent training, etc. AI allows the bot to learn from its interactions with the end-users. It can use analytics platforms, and integrations with APIs, that feed the AI and provide resources to provide the user with the most accurate and context-driven response.

4. HOW COMPLEX IS IT TO DEVELOP A CHATBOT?

Human conversations are random and complex, with endless possibilities and often come with an uncertainty of the context. A successful bot program should be able to handle difficult and complex cases or scenarios that might be unknown to it. This situation needs to be considered and addressed in advance, while programming the chatbot. The information shared with the bot may either have enough information or no information or even irrelevant information. Good chatbot programming handles all such similar user intents during the chatbot programming phase itself. An imperative consideration is to identify the kind of audience the bot is going to serve. This ensures that the conversation is designed in a manner that resonates with the user and creates an engaging end-to-end conversation with the customer.

It is important to also recognize that despite the best of efforts, some scenarios can always remain unknown. For such situations, the chatbot should be programmed to move out of complications and unknown scenarios, take a conversational/human interactive approach or just move it to a human customer representative at the right time. Such exceptions should be handled gracefully, and the user would feel more comfortable and assured with a human interaction in such scenarios.

While developing a chatbot, the following obstacles or considerations should be considered:

- Manage bot behavior with new unfamiliar data. While developing chatbots, inadequate focus on handling unpredictable situations often leads to a failed or incompetent chatbot.
- Ensure complete coverage of localization, performance & security beyond functionality
- Choose/switch between voice and text-based chatbots
- Understand the target audience for the chatbot, their requirements and anticipate their queries
- Build a cut-throat user experience that is compatible with different digital channels and platforms

- Ensure the chatbot is customized and fits in with your business and brand identity
- Integrate feedback from beta phase after continuous training and testing
- Test critical scenarios considering the uncertainty of user conversation
- Ensure domain-specific validation
- Support multi-language choice, particularly in voice-enabled chatbots
- Understand the algorithm applied to the bot and ensure NLP model testing

The only way to build a successful chatbot is by creating an interactive, relatable, accurate, and reliable bot. This is possible when you have identified the requirements besides also having a standardized and thorough chatbot testing strategy.

5. HOW CAN A STANDARDIZED CHATBOT TESTING APPROACH HELP?

Several platforms and tools are available for testing the chatbot development but one sore point with each of them is the lack of any standardized approach to chatbot testing. Chatbot testing tends to differ a lot from the traditional testing approach for a website. This is primarily because of the randomness and uncertainty in the conversation. It is almost impossible to consider every possible situation that can happen during a chatbot conversation. The bot interacts with different cloud services, different platforms and multi-channels simultaneously. Besides this, there is no limitation on user input either, because any user can add any detail in the chatbot. Therefore, a chatbot's functionality, security, performance, and exception handling should be robust.

Chatbot testing should begin with manual testing, followed by testing with a closed group of people to get real-time feedback. This can be done by collecting the data for unexpected behavior and invalid data responses. This will help mature the chatbot by augmenting its data model. Although automation testing can help in the functional aspect, there is no shortcut to test the conversational flow. Chabot testing must consider accuracy of results, ease of use, and must get the user engaged in its conversational flow.

The different types of chatbot testing that can be performed depend on the chatbot's goal and its targeted user profile. The next section will cover all the different types of testing which are possible for a chatbot.

5.1. What is the scope of testing chatbots?

5.1.1. Personality

Does a chatbot have its own personality and name that fits with its service offerings. Does the tone of the chatbot change each time the user converses with the chatbot or does it remain the same?

5.1.2. Onboarding

Does a chatbot welcome and greet the user when he/she starts the conversation? Does the chatbot mention about its capabilities and goals?

5.1.3. Understanding

Is a chatbot able to understand small talk, idioms, emojis, etc.? Does the chatbot understand the question being asked during the conversation?

5.1.4. Accuracy

Does a chatbot respond with a valid answer on being asked a question by the user? Does the chatbot respond in multiple steps?

5.1.5. Intelligence

Does a chatbot have any intelligence? Does it remember the things that have been answered by the user or asks the user again? Does it remember the context of the conversation? Does it respond when the user asks a question to it in a single step or miss one of the steps during the conversation?

5.1.6. Navigation

Does a chatbot allow the user to go back and change the context of the conversation?

5.1.7. Error Management

Does a chatbot have the capability to handle a situation where it does not have any answer to the question being asked by the user?

5.1.8. Speed

How long does the chatbot take to provide a response to the user's query/question? Does a chatbot take too much time to respond?

5.2. Types of testing

Listed below are the different types of testing that should be performed for a chatbot before going live with it:

5.2.1. Onboarding & Personality testing

Start testing the chatbot with some basic questions and check if it answers along the expected lines. Q&A should begin with the broader questions that any chatbot can answer and can get specific contextually in subsequent questions.

Onboarding and personality testing also help in establishing the onboarding process, where a user lands on a chatbot and starts the conversation. The chatbot should be able to welcome and greet the user and provide answers to basic questions. If the chatbot fails to pass this general test, it does not make sense to proceed with the next level of detailed testing.

Before a bot is designed, it is given a personality of its own. As a pre-requisite towards building the bot, a lot of thought is put around what the bot would be like, how it would respond, its tone, its sense of humor, etc. This decision on the chatbot's personality depends on the business requirements it is supposed to address. As an example, Nagarro's employee support application is an internal bot called Ginger, a customized personal assistant. It provides all information about the organization, based on roles assigned to them. Apart from this, it also provides information about all conferences, events, notifications, specific alerts, customized reports and more.

5.2.2. Conversation flow testing

Conversation flow testing identifies the different purposes that a chatbot serves and visualizes the flow for different purposes in a flow chart. This is followed by preparing the test cases for all the different flows that are possible during a chatbot conversation. It is very important to prioritize the different flows and begin the testing activity with the highest priority flows, in decreasing order of priority.

Testing can either be performed manually or by using automation solutions. For example, Azure, a LUIS-based bot can be tested manually using a webchat provided by Microsoft. Using the Botium end-to-end framework ensures conversation flow testing. Another utility is used to reduce writing conversation flows as test scripts in Botium framework by using the Knowledge Box repository.

5.2.3. NLP testing

The chatbot should be tested to ensure that it can understand the intent of the user's conversation and can respond appropriately. In case of an invalid question, it should be tested to ensure that it responds by default responses. If a chatbot supports features like sentiment analysis and speech recognition, these should also be considered during testing. For NLP testing, we need to understand the NLP model, complex algorithms, business rules, and more because this testing is the most complex aspect and demands more insights into training the bot. For NLP testing and real time monitoring of NLP model test, use QBox.ai. QBox.ai helps compare the existing version of the bot with newly tested versions and adds an alert mechanism to get timely notifications. QBox.ai gives a detailed analysis report and drills down to the exact question that requires optimization for better corrections, stability, and confidence score.

NLP testing uses automated tools to simplify in just a few clicks. All you need to do is to:

- Import the model file from the implemented platform on which the bot is created
- Upload the model file to QBox.ai
- Add the threshold for the bot, and
- Execute to get the correctness, stability, and confidence score of your bot

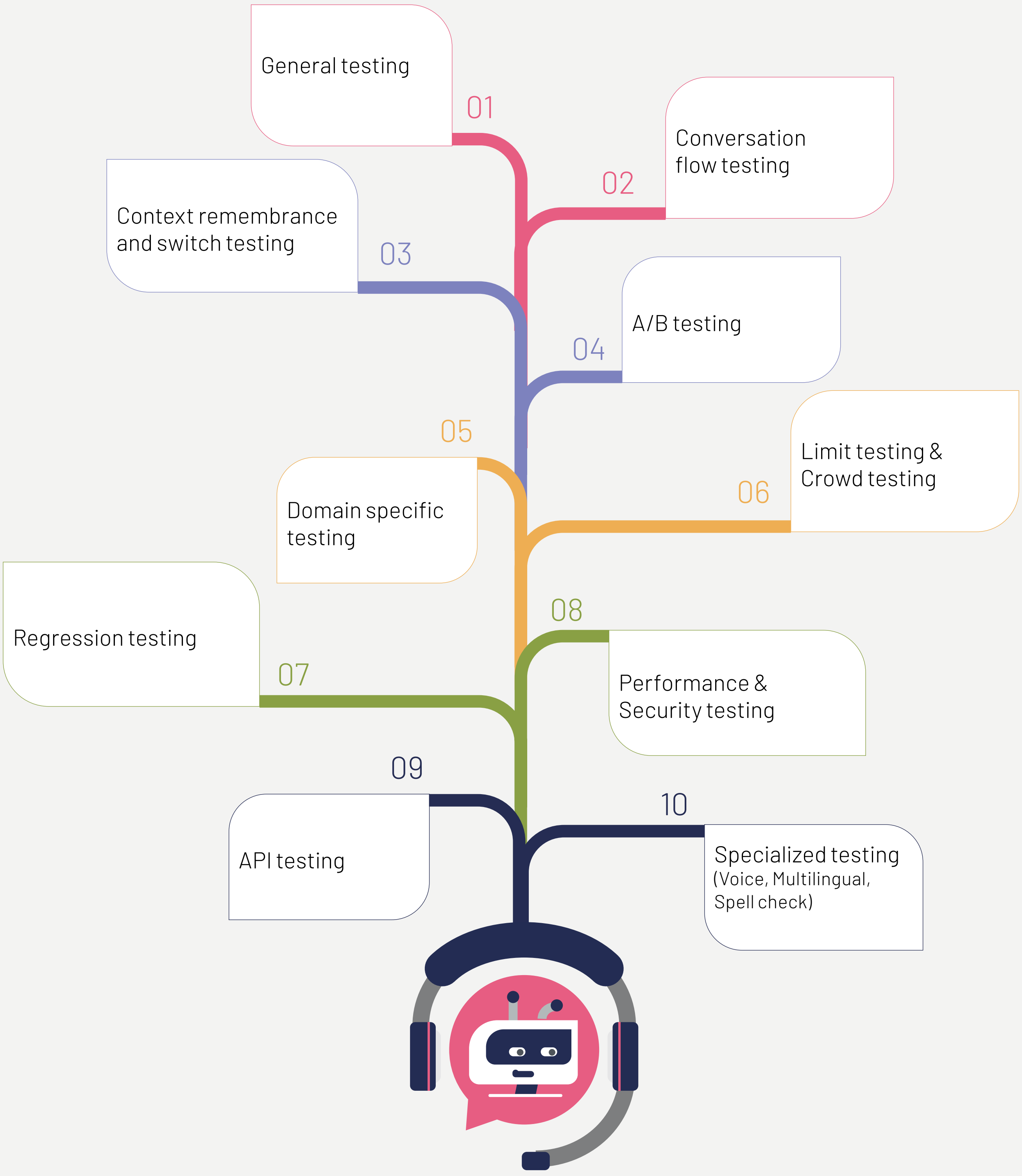

General testing
01
Conversation flow testing
02
Context remembrance and switch testing
03
A/B testing
04
05
Domain specific testing
Limit testing & Crowd testing
06
Regression testing
07
08
Performance & Security testing
09
API testing
10
Specialized testing (Voice, Multilingual, Spell check)

Figure 2: Types of testing

5.2.4. Context remembrance and switch testing

This testing identifies the contexts for the different conversation flows and then assesses if the chatbot really remembers the context or if it can understand whether the context has switched in the middle of a conversation. This testing should be done manually.

Consider a case where the user asks the chatbot to book flight tickets for 3 people from Los Angeles to New York. If, during the conversation, the user asks for flight options to California, the chatbot should remember that the user is looking for flight tickets for 3 persons from Los Angeles. Simultaneously, the user can change the context and can ask about the boarding process. In such a scenario, the chatbot should switch to the new context and respond to the conversation accordingly. Therefore, context of the conversation is very important.

5.2.5. A/B testing

The flow charts prepared during the conversation flow testing do not reflect the unlimited possibilities of a user to express an intent with different word utterances. The different utterances for each step during a conversation are identified and each flow is tested for each utterance.

As an example, consider the case where the user asks to book flight tickets for 3 members from Los Angeles to New York. The user can ask the chatbot to book tickets in different ways, like: "I want to fly from Los Angeles to New York," or "I need flight tickets from Los Angeles to New York", or "I need air tickets from Los Angeles to New York", or "I need tickets to New York from Los Angeles, etc." and so on. All these different utterances of saying the same thing of booking flight tickets should be identified and tested.

Since the list of such utterances can be endless, this testing can be automated by identifying the most common utterances, by using Qbox.ai for testing intent, entity, and utterances.

5.2.6. Domain-specific testing

A chatbot should be tested by including domain-specific keywords in the conversation flows. This testing identifies the domain of the chatbot and the relevant keywords of that domain. Once the list of domain keywords is identified, those keywords should be included in conversation flow testing. This ensures that the chatbot correctly understands them as per the domain it serves.

For example, a chatbot built for IT should treat the word "Selenium" as an automation tool while a chatbot built for the pharmaceutical industry should treat the keyword "Selenium" as a chemical.

5.2.7. Limit testing

A chatbot should be tested for invalid inputs and to see how gracefully they are handled. It would be easier to see what happens when the chatbot fails. This type of testing can be done by identifying the invalid inputs during the different stages of a conversation flow and then verifying how the chatbot would handle it.

5.2.8. Crowd testing

This testing focuses on handing over the chatbot to a small group of beta users, who can test it with a different combination of valid and invalid questions. It assesses how the chatbot behaves in real time. Crowd testing helps build human intelligence directly into the chatbot and reach a higher confidence interval. This also helps in finetuning and maturing the chatbot before going live.

5.2.9. Regression testing

Chatbots keep evolving continuously over time. It is developed and tested with the introduction of new intents and utterances added to the NLP data model. With the addition of new intents and utterances, it is possible that old intents and utterances are not recognized by the chatbot or the chatbot's confidence level reduces significantly. This type of testing focuses on running all the existing conversation flows after each change to see if the chatbot recognizes the existing intents and utterances correctly with the same level of confidence as earlier.

Regression testing should be automated because running the previous cases manually would certainly take more time. Moreover, it won't be feasible to run all the cases. Regression testing can be automated by using a combination of tools for API, conversational flow, and the NLP model.

5.2.10. Performance testing

Performance testing focuses on the time taken by a chatbot to respond to any question, the number of words returned by chatbot in its responses, and the number of steps in which a user's query is being resolved. The performance is evaluated as per the number of platform interactions and query-based responses to be measured and optimized. For a chatbot's performance testing, its three main components – the bot framework, bot service and channel tests – must be tested. Bot services can be tested using JMeter while testing the others depends on how they have been deployed.

For example, if the chatbot is implemented on Microsoft Azure using Microsoft bot framework, bot services and integration is done through the bot channel. The Microsoft internal performance testing service performs end-to-end testing. It also provides real-time monitoring with auto-scaling – a feature which can be enabled in case of heavy load. One should never use direct line to test your bot, which is provided by developers for automation conversational flow automation and to get the response from your bot.

5.2.11. Security testing

The chatbot should be tested to ensure that the privacy data shared by the user with the chatbot during a conversation is protected and cannot be used by any third party. We should ensure security testing while designing the bot, and should also understand the security framework and mitigate any risks with threat modeling.

This also includes the aspect of regulatory compliance, governance risk, and compliance & cloud management platform. Besides this, we should also test security at each level of testing and exceptional handling by using compliance coding guidelines. We must also test vulnerability assessment and penetration testing by using tools and manual security testing. Before the chatbot goes live, we must beta test the security protocol and check whether the data going on the Cloud or on-premise is secure.

5.2.12. API testing

This type of testing focuses on testing the backend services of the bot to ensure that services can be tested standalone with different kinds of inputs, to expedite the testing process without waiting for the chat interface. Since security is a major concern for chatbots, it is extremely crucial to perform security testing of the APIs that the bot relies on. Security testing should be done for APIs to ensure that they are free from common vulnerabilities.

This type of testing should be automated with the different tools available in market for API testing.

5.2.13. Multilingual testing

This type of testing should be done by expert users who are well-versed in the supported language. This type of testing should be avoided by using different online language translators as these tools are not very concise while interpreting the user sentiments and emotions.

5.2.14. Voice quality testing (listening and speaking)

In today's times, chatbots also come with voice support in the form of voice to text (Siri, Cortona), and voice-only (Alexa, Google Home). The initial round of voice testing is performed by testers to see how the chatbot understands voice. Once initial voice testing gives satisfactory results, the chatbot can be handed over for crowd testing. Crowd testing involves having users with different accents to see how the chatbot understands them. It is also important to check for homonyms (words which sound identical, like sun and son) too during voice testing. A chatbot engine should be smart enough to finetune itself to the user's voice if the same user talks to the chatbot multiple times.

Voice testing can be done manually. For manual testing using voice, we can use any echo device or virtual solutions and simulators. End-to-end automation testing is performed by using scripts with components which allow interactions with voice bots, without speaking programmatically. Continuous testing and monitoring are the last frontiers for voice testing, to check the test skill on regular interval and to notify whenever any such issue arises.

5.2.15. Spell check testing

It is quite possible to make a typographical/ spelling error in the text entered by the user while performing conversational flow testing. Chatbots should be able to understand the misspelt word corresponding to the entity name, by training themselves with synonyms for entity name, including misspelled words which are similar to the entity.

6. BEYOND TRADITIONAL TESTING FOR AI-INFUSED CHATBOTS: KEY CONSIDERATION

6.1. End-to-end testing and release process

Ideally implemented after functional & system testing, end-to-end testing is a reliable method to assure that the flow of an application is performing as designed & as intended right from the start to the finish. This will help quality experts identify system dependencies & guarantee that the right information is transferred from various system components & systems. Fig. 3 illustrates how to test end-to-end conversational flow, natural language understanding & self-improvement attributes.

6.2. Measure the effectiveness of your chatbot

To evaluate a chatbot's effectiveness, the following metrices should be measured. These identified metrics are a comprehensive toolset that provide value to the users and help track the overall performance of the chatbot.

NLP provides automatic simulation of a natural language that resembles speech and text. It is a subfield of computer science, concerned with the interactions between computers and human languages. NLP is all about creating interactions between computers and the users by understanding inputs. This is achieved by translating them into their known language.

6.2.1. Comprehension capabilities

Good comprehension capabilities of a chatbot ensure a good texting and error-free experience for the user. When a user makes a spelling mistake or makes an error in a sentence, the chatbot should enable the 'auto-correct' feature. The Bing Spell Check API from Microsoft's Cognitive Services is the best example of comprehension capabilities. This cognitive service adds intelligence to chatbots with spell check capabilities like 'Spell' for web searches, to return better search results for the user.

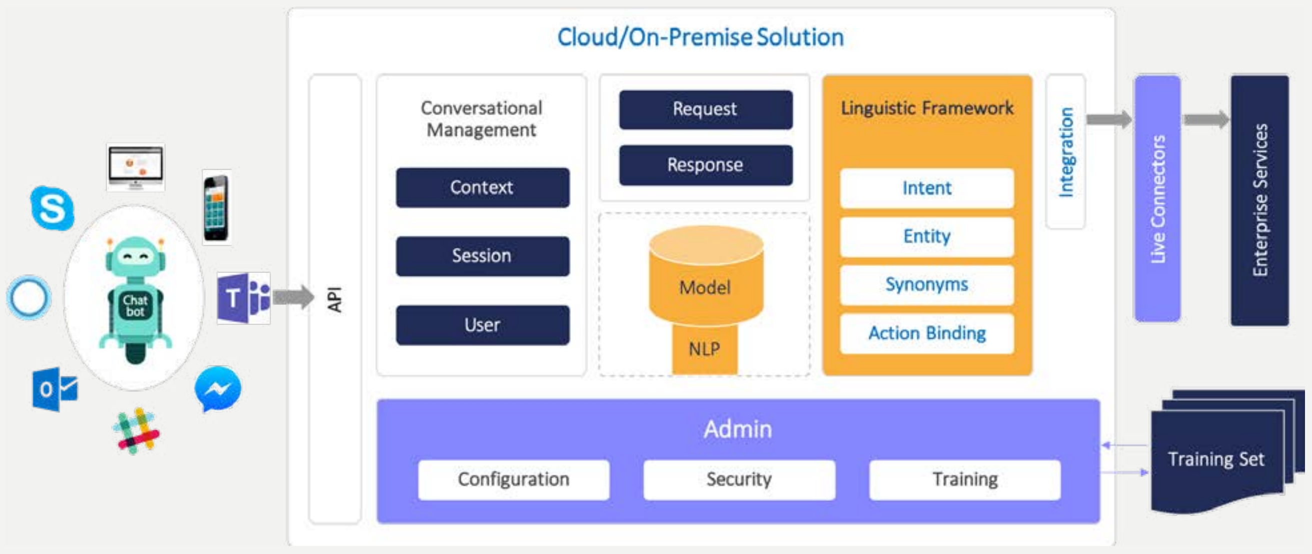


Figure 3: End-to-end automation testing framework

Further, chatbots should also have two types of intent understanding abilities:

1. Text-based understanding - to quickly understand the intent/departments/entities behind the questions and statements the user is texting.

2. Chatbots should be capable of 'balanced text-use'. This means that a chatbot will use the combination of both short descriptions and engaging content like rich media to hold the user's attention.

6.2.2. User engagement

Good chatbots should be capable of initiating a conversation with the users and interacting with them to share information. Apart from this, chatbots should be built to classify the target audience, deliver meaningful messages, take direct orders from users, and navigate layouts. Chatbots should also be designed to answer frequently asked questions (FAQs) of users, by accessing personal information, account status, purchase history, previous actions, and more. These user-engaging attributes lead to a good user retention rate.

6.2.3. Speed

One of the prime intentions for the existence of a chatbot is to help the users instantly, thanks to their instant response mechanism. When building a chatbot, it is ensured that they are integrated with knowledge-based database and programmed to fetch information and respond quickly. This leads to prompt and effective interactions with the users.

6.2.4. Functionality

Good chatbots can be created through a variety of well-designed functionalities such as onboarding, rich media use, and navigation that lead to a great conversational flow. For instance, chatbots should welcome users with a series of onboarding steps. They can also use engaging rich media images with text to capture the user's attention and to provide navigation tools to help the users with the layout.

6.2.5. Interoperability

Interoperability simply means the ability of computer systems or software applications to exchange and make use of information. A

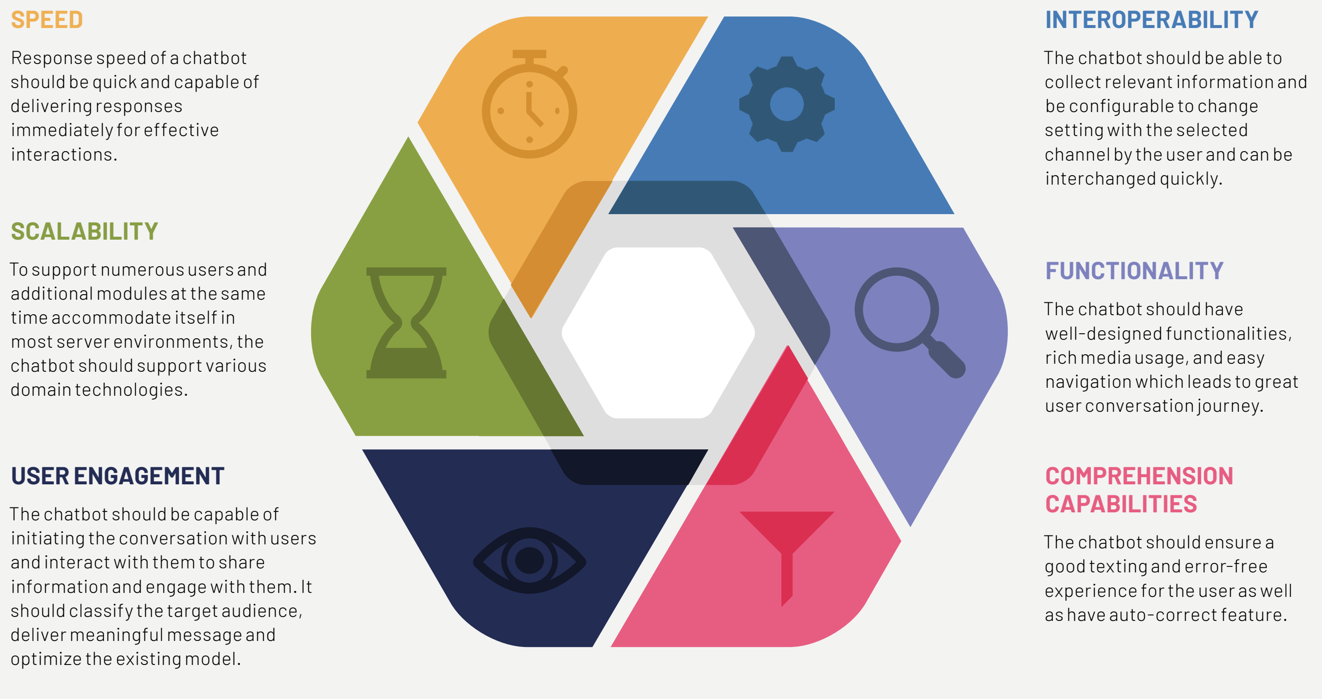

SPEED
Response speed of a chatbot should be quick and capable of delivering responses immediately for effective interactions.
SCALABILITY
To support numerous users and additional modules at the same time accommodate itself in most server environments, the chatbot should support various domain technologies.
USER ENGAGEMENT
The chatbot should be capable of initiating the conversation with users and interact with them to share information and engage with them. It should classify the target audience, deliver meaningful message and optimize the existing model.
INTEROPERABILITY
The chatbot should be able to collect relevant information and be configurable to change setting with the selected channel by the user and can be interchanged quickly.
FUNCTIONALITY
The chatbot should have well-designed functionalities, rich media usage, and easy navigation which leads to great user conversation journey.
COMPREHENSION CAPABILITIES
The chatbot should ensure a good texting and error-free experience for the user as well as have auto-correct feature.

Figure 4: Parameters to evaluate chatbot effectiveness

well-designed chatbot should be deployed in such a manner that it can support multiple channels such as Teams, Skype for Business, and web browsers. Users should be allowed to quickly change the settings to run the chatbot on any selected channel. This allows users to get maximum search results from various channels.

6.2.6. Scalability

Good chatbots should be scalable enough to support numerous users and additional modules concurrently. A chatbot should be built to accommodate itself in most server environments as per the various industry requirements. A chatbot should be capable of working on either of them, irrespective of server environment.

6.2.7. Key Performance Indicators in a chatbot

To measure the performance and capabilities of chatbot, it is very important to first identify and measure the metrics which can help in identifying the opportunities to improve the chatbot. Figure 5 shows the key metrics that can be considered as the key performance indicators in a chatbot.

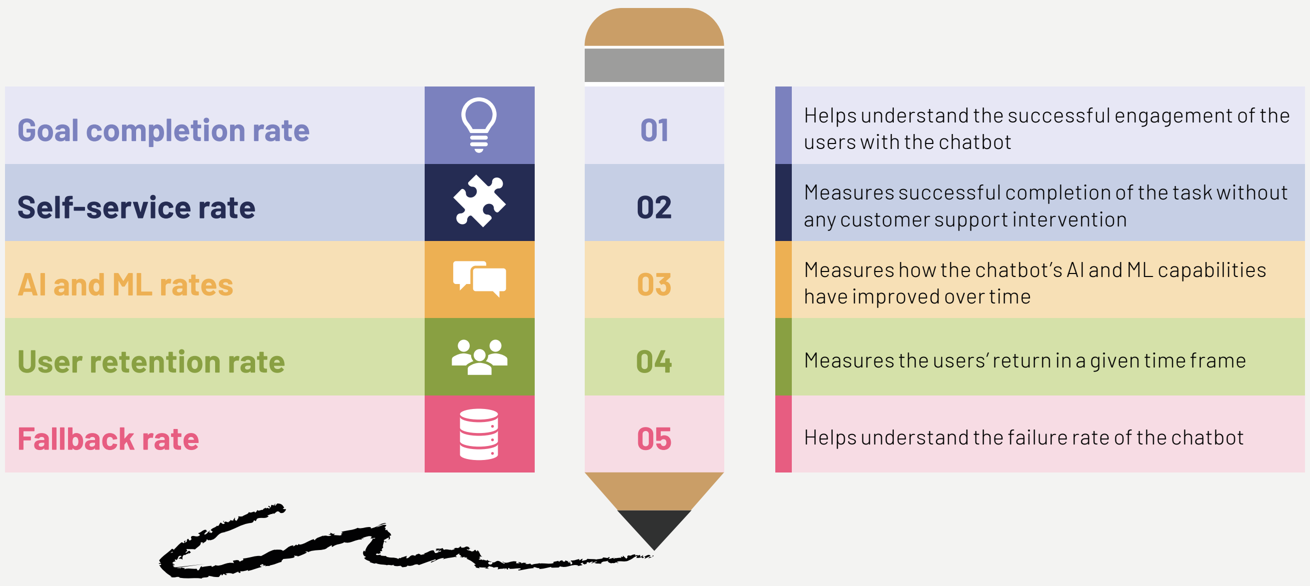
6.3. Why should chatbots be based on NLP?

6.3.1. Intent recognition

1. NLP can break down chatbot tasks to a few words, to find the meaning of each word. It can recognize the intention of the customers by analyzing the typical human command structures. NLP accepts each word to interpret the synonyms of evident and non-obvious words correctly.

2. NLP intent recognition not only matches the utterances with the task and but also with the intended function. This is done in the following ways:

   - Differentiating numeric words from digits
   - Removing capitalization from common nouns by organizing recognizing proper names
   - Personalizing messages, replacing default and universal with uniquely configured customized messages



Goal completion rate	01	Helps understand the successful engagement of the users with the chatbot
Self-service rate	02	Measures successful completion of the task without any customer support intervention
AI and ML rates	03	Measures how the chatbot's AI and ML capabilities have improved over time
User retention rate	04	Measures the users' return in a given time frame
Fallback rate	05	Helps understand the failure rate of the chatbot

Figure 5: Key performance indicators in a chatbot

"The limits of my language are the limits of my world."
- Ludwig Wittgenstein

- Making continuous chatbot vocabulary expansion possible by using ML for adding synonyms, and by also including pre-programmed synonyms for both types of responses
- Developing vocabulary transfer from one chatbot to another is made possible
- Removing apostrophes and expands contractions to simplify processing of tasks
- Enabling same way process of singular and plural nouns

3. Understanding single verbs communicated in different tenses

6.3.2. Entity extraction

NLP allows developers to designate entities or the fields, data, or words that are necessary for a chatbot. It enables the chatbot to complete tasks as per date, time, person, descriptions of products, and other criteria. You can do so by identifying the user's utterances across all fields to match the task at hand. It also collects additional field data as per requirements.

NLP recognizes the critical nuances of a human's natural language and mitigates potential misinterpretations. It helps chatbot developers to customize, expand, and re-use vocabulary and every natural variation easily. The goal is to complete the task by filling in the gaps and by ignoring unnecessary details through a deductive process.

6.3.3. Overwhelm language barriers

"The limits of my language are the limits of my world." - *Ludwig Wittgenstein*

Language barriers should not serve as any limitation to restrict businesses. In today's day and age, businesses regularly rely on translation of text and voice. NLP-based chatbots can easily interpret messages despite any mistake.

6.3.4. Understanding morphemes & slang of different languages

NLP enables chatbots to understand morphemes and abbreviations across different languages. It also allows learning slang that is similar to a human and makes chatbots more believable, with fewer errors.

6.3.5. User experience

NLP fulfills customer requirements with better clarity and accuracy. It has become popular in businesses because of its ability to provide a better user experience across applications like chatbots, social media listening and texting, and voice-enabled applications and devices.

In this globalized world, NLP's translational and other features extremely valuable and sought-after. Which is why, NLP-based chatbots improve customer experience. You can fulfill the expectations of today's millennial users around the globe with NLP's business intelligence.

6.4. Identify advanced chatbot security threats

There are two types of risks/hazards to chatbot protocols that one should guard against:

Threats

Threats are one-off events, such as malware or DDoS attacks. Business-specific targeted attacks can lock you out of your system and can hold it hostage. Alternately, hackers can threaten to expose (ostensibly) secure customer data. Threats rely on systems being vulnerable to attack because of poor security such as weak firewalls or open networks. Your must ensure that the firewalls are robust. The first layer of defence should deter most attacks before they get any deeper into the system.

Vulnerability

A system can be compromised because of weak coding, poor safeguards, or the weakest link in the chain, user error, etc. All systems have weak spots – no system is entirely 'hack proof'. However, chatbots can present a solution to these weaknesses. Their algorithms are so complex that they reduce the vulnerability of a system through a web of integrated security protocols. Thus, not only can a good chatbot be an effective interface between businesses and their customers, it can improve their security as well.

6.5. Overcoming threats and vulnerability by securing chatbots

While chatbots are relatively new, the protocols, the systems and the coding used to protect them are almost identical to those used in existing HIMs. They interact across platforms that already have their own internal security systems. This means that from the very outset, users are protected by multiple layers of encryption and security.

The techniques developers can use to protect the system include:

- Biometric authentication: Iris scans and fingerprint scans are increasingly popular and reasonably robust, thanks to advancements in biometrics technology.
- Two-factor authentication: Users are required to verify their identity through two separate channels. Despite being 'old school', sometimes such tried-and-tested methods are the best form of defense. Two-factor authentication is still used by many financial and banking institutions.
- User ID: The most familiar method of security to the average digital customer, user IDs involve creating secure login credentials, including passwords that are not their pet's names or just the word 'password'.
- Authentication Timeouts: A 'ticking clock' for correct authentication input can prevent repeated attempts to try and guess their way into a secure account by hackers.
- End-to-End encryption: This stops anyone other than the sender and recipient from seeing any part of the message. This could be incorporated into chatbots very soon and is undoubtedly one of the most robust methods of ensuring chatbot security.
- Other methods could include 2FA, behavior analytics, and even the continuing evolution of AI itself.

Despite all the advanced infrastructure, coding and guidelines that we may follow, there is always speculation around the efficiency of chatbot security. At Nagarro, we abide by the following actions to enhance chatbot security (these are also considered during testing and while using static and dynamic security testing tools, and in API testing):

1. Ensuring privacy encryption of customer data
2. Validating network security
3. Verifying the highest level of authorization protocol
4. Checking for authentication time out
5. Validating the compliance guideline
6. Verifying security threat at each integration level

7. WHAT ARE THE SUITABLE CHATBOT TESTING TOOLS?

Only a few tools available in the market can help in automation testing of the chatbots. The following testing tools can help you with automation testing of different conversation flows and NLP data models:

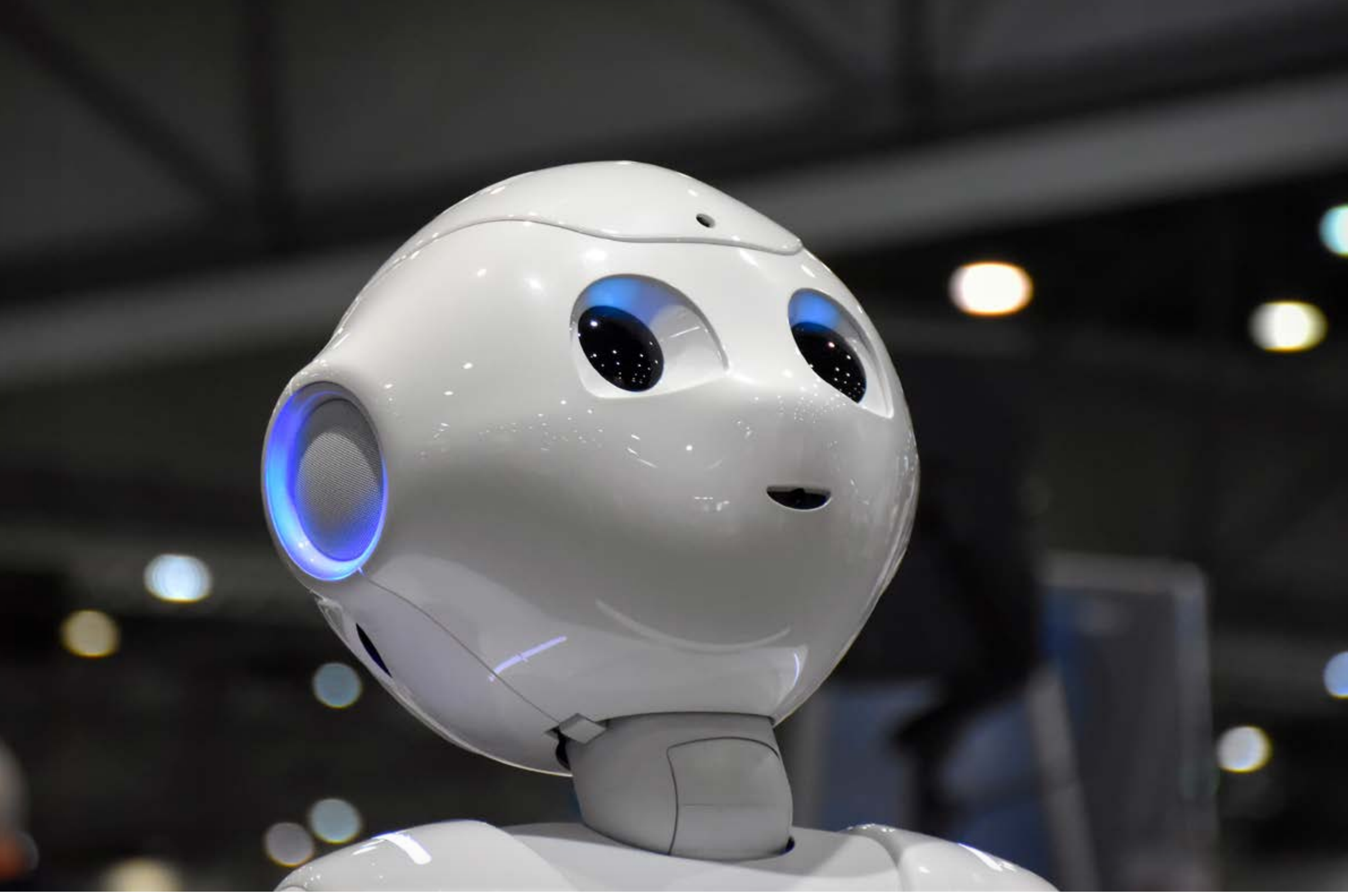
1. Chatbottest is an open source guide that helps you identify any issues with chatbot design under seven different categories - Personality, Onboarding, Understanding, Answering, Navigation, Error management, and Intelligence.

2. Botium is a test automation tool to automate the conversation flows.

3. TestMyBot is a test automation library for chatbot conversations. It includes tools for recording and replaying conversations and integrates with CT/CI/CD pipelines(Continuous Testing, Continuous Integration, Continuous Delivery).

4. Qbox.ai is an automation tool that helps to quickly and easily understand, analyze, and improve the performance and results of chatbots and conversational AI platforms.

5. Out of these testing tools, Qbox and Botium are the most popular tools for automation. QBox helps in testing the NLP model and Botium helps in test conversational flow, which is easily integrated with Selenium.



CONCLUSION

Gartner predicts that by the end of 2019, 40% of enterprises will be actively using chatbots to facilitate business processes by using natural-language interactions. Many industries across the globe like healthcare, e-commerce, banking, financial, IT, customer service, retail, etc., are adopting artificial intelligence (AI)-powered chatbots to automate a range of tasks and simplify business processes. Entering chatbot development is still easy and quick. However, to accomplishing today's AI-based bot demand and testing these automated bots with diverse technologies and platforms is very difficult. To some extent, we have covered all the testing areas to ensure our chatbot can work as required and as expected. A human-centric approach to AI can make all the difference. The key to successful bot testing is to have a continuous train-and-test approach.

ABOUT THE AUTHOR

Rajni Singh has worked in nearly every software development role from development, testing and DevOps to security, performance, and program management. Her expertise lies in developing automation frameworks with high quality services, performance engineering and developing strategy for emerging technologies such as Blockchain, conversational AI, IoT, AR/VR and more.

Neeraj Jain has total 12+ years of experience in software testing with specialty in advanced test automation, performance engineering and has deep understanding of testing concepts and strategies ranging from traditional to emerging technologies such as AI/ML, chatbot, Blockchain, and more.

nagarro
THINKING BREAKTHROUGHS

ABOUT NAGARRO

Nagarro drives technology-led business breakthroughs for industry leaders and challengers.When our clients want to move fast and make things, they turn to us. Working with some of our SAP clients, we continuously push the boundaries of what is possible related to innovations through process optimization and technology progress. Today, we are more than 6000 experts across 21 countries. Together, we form Nagarro, the global services division of Munich-based Allgeier SE.

CONNECT WITH US

info@nagarro.com www.nagarro.com /nagarroinc /nagarro /nagarro /nagarroinc /lifeatnagarro