Digital Operations

The Nuts and Bolts of Bot Engineering

These six engineering principles can mean the difference between robotic processing automation success and failure.

Executive Summary

Companies in every major industry are turning to semi-autonomous computers (better known as bots) to automate large- and small-scale business processes. This technology replaces human intervention in back-office operations, improving operational efficiency, reducing costs and increasing margins.

For most large organizations, the question is not if they will employ robotic process automation (RPA) software, but when. Data points from top industry gurus say it all:

- "By 2019, 35% of leading organizations in logistics, health, utilities and resources will explore the use of robots to automate operations." – IDC
- "Robotic process automation software market will grow by 41% year over year to 2020." – Gartner
- "By 2021, there will be over 4 million robots doing office and administrative and sales and related tasks." – Forrester Research

However, organizations cannot employ this technology effectively unless the bots are engineered properly. This white paper provides an overview of the engineering principles that every company needs to follow to achieve maximum automation with near 100% accuracy. These principles include architecture and design; capacity planning and bot optimization; coding best practices; fail-proof unit testing; quality assurance (QA) and user acceptance testing (UAT); and real-time monitoring and reporting.

Cognizant

Key challenges in RPA bot engineering

When evaluating bot engineering, organizations must prepare their IT business application landscape to accommodate bots. Common conditions or scenarios they often face include:

- IT applications are slow.
- There are network or infrastructure issues, such as network downtime.
- Changes to business applications are not aligned with new software releases.
- The dynamic behavior of screens, such as a refresh, interrupts bot execution.
- There are unexpected exceptions in the applications.

The bot engineering lifecycle

All software engineering follows a design, development and maintenance process, and development is performed in stages that follow the software development lifecycle (SDLC).

Accordingly, bot engineering is conducted in a staged approached, as illustrated below (see Figure 1).

The bot engineering lifecycle

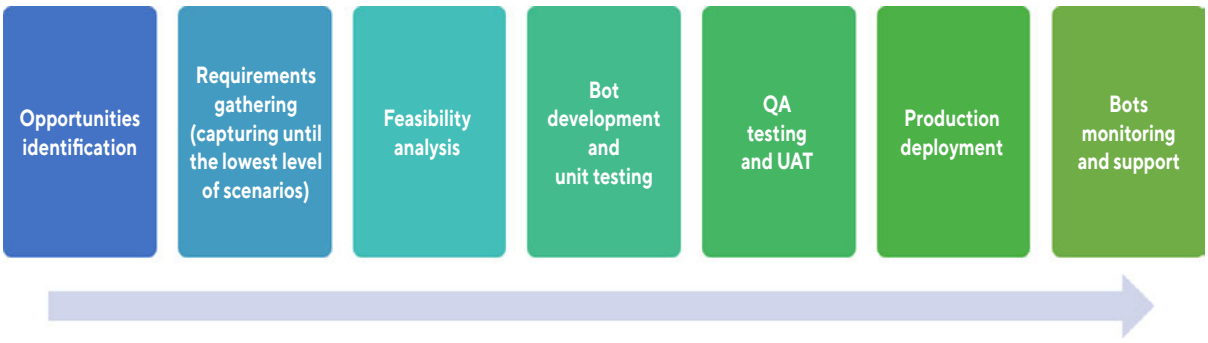


Figure 1

The six engineering principles

All commercially available RPA tools provide a platform for developing automation scripts, with features for any technology used by the underlying business applications.

To build reliable, error-free bots that will perform automated tasks for any business application, companies should follow the following engineering principles.

1. Architecture and design.
2. Capacity planning and bot optimization.
3. Coding best practices.
4. Fail-proof unit testing.
5. Quality assurance (QA) and user acceptance testing (UAT).
6. Real-time monitoring and reporting.

Architecture and design

There are two key architecture and design principles that engineers must follow regardless of what kinds of bots they are building.

- **First, set up a framework to capture exceptions and log errors.** As bots are built to perform rule-based tasks in the automation workflow, the framework should be designed to capture any exceptions expected in a particular business process automation flow (See Figure 2).

Using custom-built frameworks across each RPA tool will enforce the design principle that should be followed by all developers. Fewer exceptions will yield a higher percentage of transactions successfully automated. Using configurable retry and rerun logic, as enforced by the framework, will help to improve overall bot performance and yield percentage.

Exception and logging framework pie chart

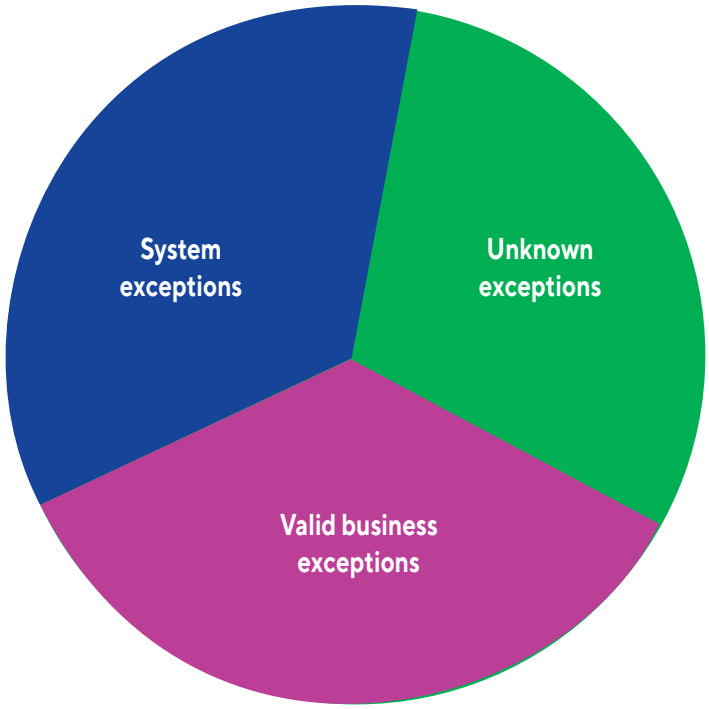


Figure 2

- **Second, bot engineering should also employ the principles of reusability and modularity.** Before developing a solution, a review of existing code should be conducted and candidates for code reuse should be identified and evaluated. Each RPA tool has features that allow your organization to build reusable components.

With respect to modularity, software should be written in a way that allows for efficient reuse by developing application programming interfaces (APIs) for common functionality. Reusable components can be built as logical bots, application-specific bots and process-specific bots (See Figure 3).

- **Logical bots:** Non-application, non-tool-specific bots can be used across any tool for any application, and can be used for any RPA tool.
- **Application-specific bots:** Screen-specific bots that can be used in any process that uses the same business application. These components are RPA-tool specific.
- **Process-specific bots:** These tools create process-specific templates that can be used across the same domain. Similar templates can be built with other RPA tools applying the same design principle.

Reusable components in building bots

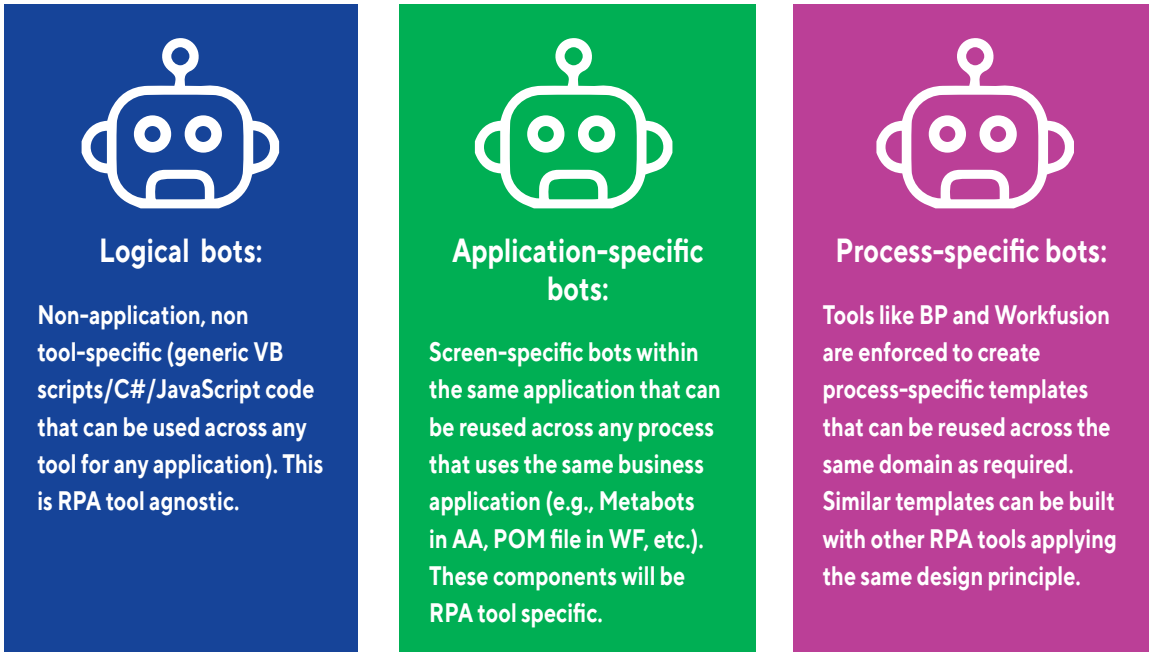


Figure 3

Capacity planning and bot optimization
In bot engineering, capacity planning needs to be conducted during the design phase to plan the number of bots required to process a given volume of transactions.

Key factors in capacity planning include:

- Volume of process transactions.
- Average handle time (AHT) of the current manual process and expected bot performance.
- Service-level agreement (SLA) for the process.
- Current process execution time period.
- Bot run-time environment and the corresponding hardware and software specifications.

With this information, your organization will be able to calculate the number of bots required to run the automated flow of the business process.

Bot optimization
Many RPA platforms have components that are priced based on required bot run-times. Therefore, it's important to complete a bot optimization exercise during the architecture and design phase. Optimization helps in bot utilization and reuse, and it is an important step in doing a cost benefit analysis to calculate return on investment.

Bot optimization is an exercise that includes the following activities:

- Identification of bots that can run in parallel.
- Scheduling of dependent bots.
- Reusable bots across processes and their scheduling time.

Coding best practices
Most RPA vendors provide specific best practices for their tools. Beyond that, there are a number of coding best practices that can be applied across any RPA tool:

- Proper exception handling should be implemented with configurable retry.
- Appropriate error logging must be implemented to capture error messages with screenshots for quicker analysis.
- Use smart/configurable delays.
- For a given process, avoid traversing to the same screen repeatedly. Read the values once and use them at all places as required.
- Avoid using the same variable name in parent and subtask if the variable is not required to be passed in subtask.
- Check if the flag is reset properly at the required places.

Defensive coding
The techniques of defensive coding, in which you ensure that your code works in all negative scenarios, are essential to achieving error-free results in bot engineering. The reason: Code variations inevitably occur due to unexpected popups, errors in the business applications, new releases, or changes in the applications or their underlying operating environment.

Examples of defensive coding principles include:

- Check for intended screens while entering or processing data.
- Verify value in dropdown after the value is selected. If the incorrect value is selected, perform a retry.
- Perform additional domain validations such as checking impacted screens/associated values. For example, after entering a policy number, check if the policy holder's name matches with the record, especially while updating or inserting any record, to ensure the accuracy of bot processing.

Fail-proof unit testing
Just as it is important to verify that the robot performs its basic functions as intended, it is equally important to verify that the robot can handle an abnormal situation. That is why quality engineering is an important step in bot engineering. The more you focus on a bot's quality before deploying it in production, the less your team will need to rely on a quality check effort by business process subject matter experts.

Testing negative scenarios during unit testing will yield more stable and reliable automation, and prevent bots from failing when they enter production and encounter invalid data. This can be done by providing invalid or incorrect data when testing bots, as it helps in discovering loopholes in the automation script.

The following are common negative scenarios that can be tested:

- **Search by providing incorrect data.** Test the automation script by providing incorrect data, such as wrong account number or policy ID, and then perform a search operation.
- **Alter screen loading delay time.** Reducing the delay time will lead the bot to the wrong screen for data entry or data extraction.
- **Capture wrong object.** Change the criteria for object capture so that the bots fail to identify the right object.
- **Hardcode variables.** To enforce negative scenarios, "hardcode" the value of variables and validate the results.
- **Introduce incorrect path.** Introduce incorrect log file paths or screen shot paths used during exception handling.
- **Process huge files.** Open applications such as Microsoft Excel or a PDF with a very large file size to mimic non-responding scenarios.
- **Simulate retry loops.** Retry testing with hardcoded values to start the retry loop again. Additionally, test when the retry loop success flag is not reset properly.
- **Introduce dynamic data population.** Introduce the wrong conditions or values in a table or grid to cause the automation to fail. In case of dropdown, select incorrect values.

QA and UAT testing

While the developers can test positive and negative scenarios during unit testing, it is essential to also complete end-to-end business process flow testing covering all the straight-through and exceptional scenarios. It is equally important to have a test environment for the QA analysts to test all the scenarios described previously. Plus, production data needs to be replicated in the QA environment to cover all scenario testing.

- **QA test data management solutions.** These can be used to create the data applied in the testing environments. For these solutions, you can utilize individuals from the IT testing team or a business operations analyst who has the required process knowledge to conduct effective QA testing of bots.
- **Similarly, it is important to have a UAT testing environment with data covering all production scenarios.** An appropriate ramp-up period needs to be defined and tested within the UAT phase. Post ramp-up, UAT testing should be established in the production region to monitor the bot's performance with increased load.
- **DevOps/continuous integration and production deployment.** Each RPA tool provides its own control room for deploying code. Here are some best practices for deployment:
  - **Continuous integration.** A defined DevOps model using additional scripts and APIs (in addition to control rooms) can be used for the smooth movement of deployment packages across different environments.
  - **Use software configuration management tools** like TFS, SVN and GitHub to enable continuous integration.
  - **Maintain a production run book** with a checklist of instructions, and production environment and machine configurations details for future maintenance.

Real-time monitoring and reporting

Building a command center/reporting dashboard-based solution will help to monitor bots in real time. This will help in analyzing system and business exceptions, and in analyzing the root causes of the issues. All RPA tools provide additional plug-ins or capabilities to integrate with dashboard tools. Such tools include Bot Insights from Automation Anywhere, UI Path-Kibana integration, Splunk-tool integration from Blue Prism, and Workflow-Tableau integration.

Any external reporting dashboard can be integrated with the RPA platforms for real-time monitoring and support (See Figure 4).

The dashboard will help to verify the automation yield percentage achieved, the number of system exceptions, and the number of business exceptions at any point during bot execution.

For example, a spike in system exceptions can help us find an underlying IT application issue, or new patches or releases. Similarly, a spike in business exceptions can help identify the business process change, or any missed business requirements.

Analyzing these exceptions in real time enables you to identify issues and act quickly, and to sustain the automation yield percentage.

Real-time monitoring and reporting dashboard

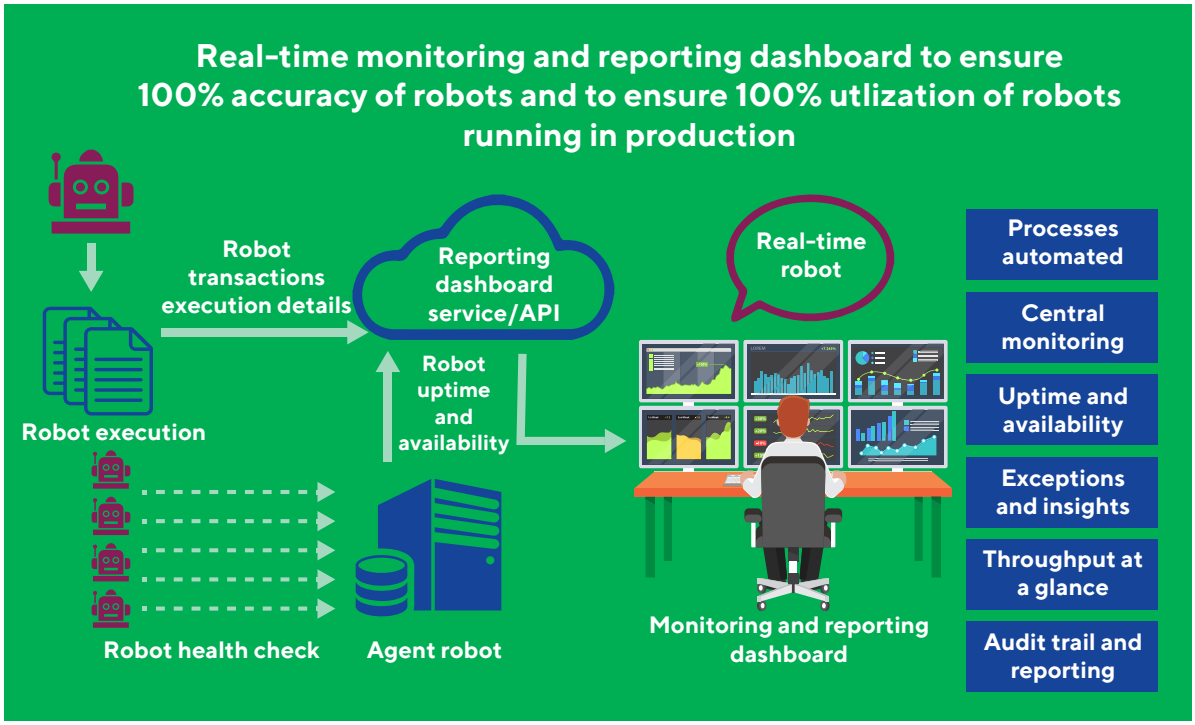


Figure 4

Bot engineering in action

We have extensive experience in both RPA and Intelligent Process Automation, which incorporates cognitive technologies.

The following are case examples of engagements in which we helped automate two critical processes for a leading U.S. insurance company. The processes involved more than 80 unique business scenarios and seven business applications. By using iterative Agile methodology, we helped achieve the targeted automation yield percentage with almost 100% accuracy.

Project-one objective:

- Eliminate repetitive tasks in processing commercial business policy renewals.
- Avoid manual errors when calculating premiums, for instance, improving accuracy.

Challenge

- Retrieving and comparing data across multiple renewal applications is done manually, making the effort labor-intensive.
- Because it is done manually, the renewal premium calculation is prone to error.

Solution

- Bots are introduced to process policy renewals, calculate pricing using a pricing tool, close the transaction and notify the customer upon completion.
- Adopted an exception and logging framework for quicker data analysis, improving the automation yield percentage and achieving near 100% accuracy.
- Embedded reporting and tracking of transactions to ensure a zero-risk environment.

Benefits

1,500+

Policies renewed weekly with an average of 300+ per day

99.9%

Accuracy

80%

Volume automated

Project-two objective:

- Eliminate repetitive tasks in commercial business policy renewals, such as premium calculations.
- Avoid manual errors and improve accuracy.

Challenge

- Multiple scenario-based verifications, which differ across states, makes the effort labor-intensive.
- When new auto policies are verified, they must be compared with prior carrier information. If there is a discrepancy, the company must request proof via email or fax, and the company must follow up to ensure that the task is completed.

Solution

- Bots were introduced to retrieve policy verification details from multiple systems to prior carrier details.
- Developed using a solid/dry principle, with a robust error handler, which ensures that new scenarios will be managed efficiently.
- Integrated with Splunk, which provides automatic tracking and alerts in the event of errors or exceptions.

Benefits

600+

Policies renewed weekly with an average of 300+ per day

99.8%

Accuracy

65%

Volume automated

Endnotes

1 "IDC Unveils its Top 10 Predictions for Worldwide Robotics for 2017 and Beyond - 2016."

2 Forecast Snapshot: Robotic Process Automation, Worldwide, 2016.

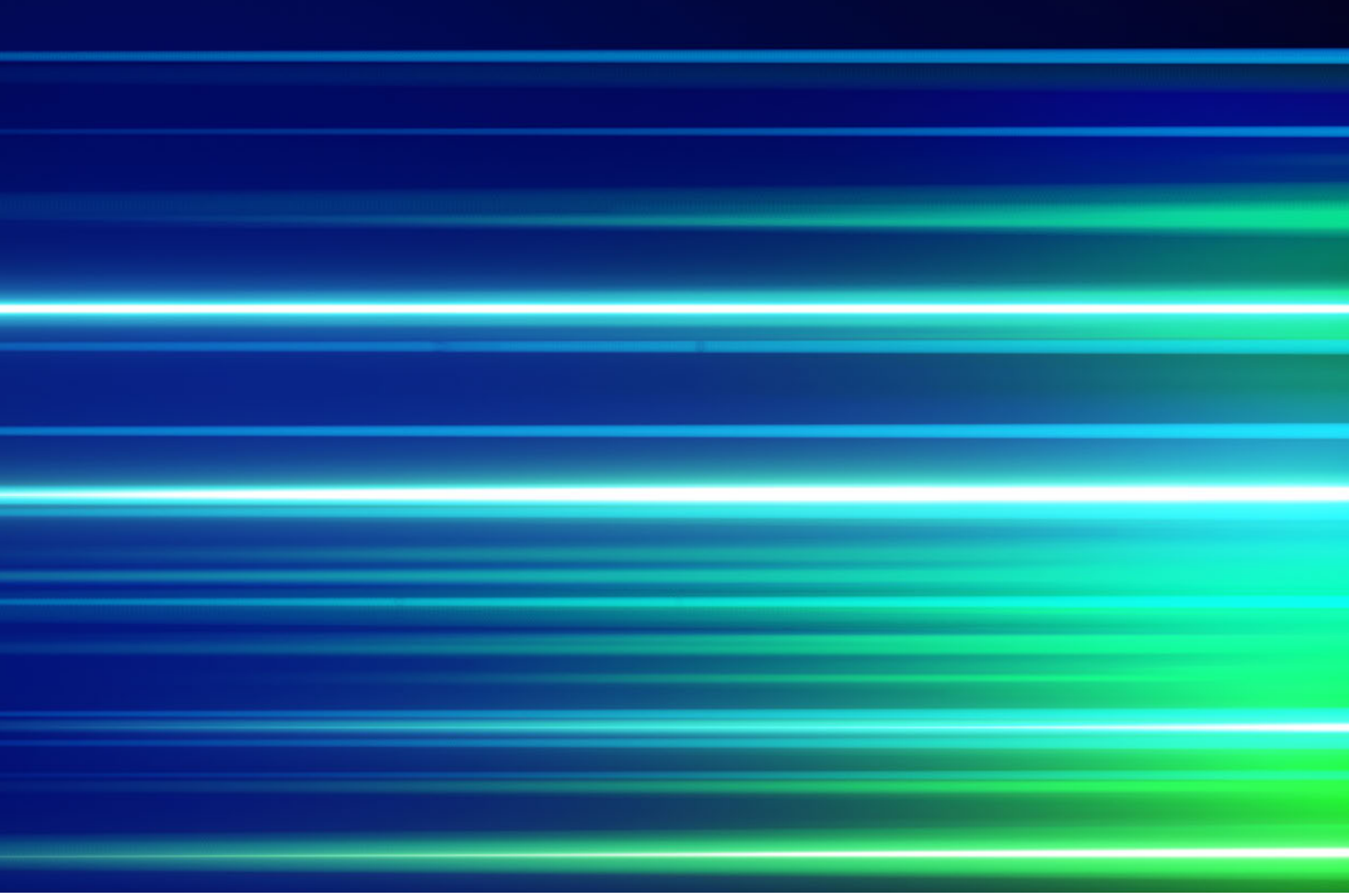
3 The Forrester Wave™: Robotic Process Automation, Q1 2017.

About the author

Vijayashree Natarajan

Cognizant Intelligent Process Automation Solutions Leader for Healthcare, Cognizant

Vijayashree Natarajan is a Cognizant Intelligent Process Automation Solutions Leader for Healthcare. She leads large automation delivery engagements using robotic and cognitive technology-based solutions, enabling digital transformation in an efficient and effective way across digital operations. Vijayashree holds an Engineering (B.Tech) degree in Electronics from Madras Institute of Technology, Anna University, Chennai, India. Vijayashree can be reached at **Vijayashree.Natarajan@cognizant.com.**



Digital Operations

Cognizant Digital Operations helps clients re-engineer, digitize, manage and operate their most essential business processes, lowering operating costs, improving user experiences, and delivering better outcomes and topline growth. Across the practice, we are creating automated, data-driven platforms and industry utilities. We help clients run better by applying traditional optimization levers, and we help them run differently by creating competitive advantage through making their processes digital-ready, which often leads to more effective operating models and corresponding topline revenue growth. Visit us at cognizant.com/cognizant-digital-operations.

About Cognizant

Cognizant (Nasdaq-100: CTSH) is one of the world's leading professional services companies, transforming clients' business, operating and technology models for the digital era. Our unique industry-based, consultative approach helps clients envision, build and run more innovative and efficient businesses. Headquartered in the U.S., Cognizant is ranked 195 on the Fortune 500 and is consistently listed among the most admired companies in the world. Learn how Cognizant helps clients lead with digital at www.cognizant.com or follow us @Cognizant.

Cognizant

World Headquarters

500 Frank W. Burr Blvd.
Teaneck, NJ 07666 USA
Phone: +1 201 801 0233
Fax: +1 201 801 0243
Toll Free: +1 888 937 3277

European Headquarters

1 Kingdom Street
Paddington Central
London W2 6BD England
Phone: +44 (0) 20 7297 7600
Fax: +44 (0) 20 7121 0102

India Operations Headquarters

#5/535 Old Mahabalipuram Road
Okkiyam Pettai, Thoraipakkam
Chennai, 600 096 India
Phone: +91 (0) 44 4209 6000
Fax: +91 (0) 44 4209 6060

Codex 3917