LTIMindtree

Sustainable Automation Solution Best Practices

Wagon Wheel Model

by Navin Singh

future, faster. Together.

Automation helps to optimize cost and maximize output, resulting in profit with minimal to zero errors. It helps achieve maximum business benefits. There is a rise in businesses investing in automation solutions, but an inappropriate solution would cost financial loss, effort, and precious time.

While there are many positive benefits of intelligent automation solutions, it is absolutely imperative that we follow the best practices and guidelines to create or choose the right solution. It is critical that we follow those guidelines throughout the life cycle to make an effective, successful, and sustainable solution. It is advisable to implement best practices that are lacking in typical processes before going ahead with the development and implementation of an automation solution.

I have derived a sustainable automation solution best practices wagon wheel model to describe critical factors that can be followed throughout the life cycle of the automation solution. It would help in building sustainable solutions and delivering required value to the targeted business entities.

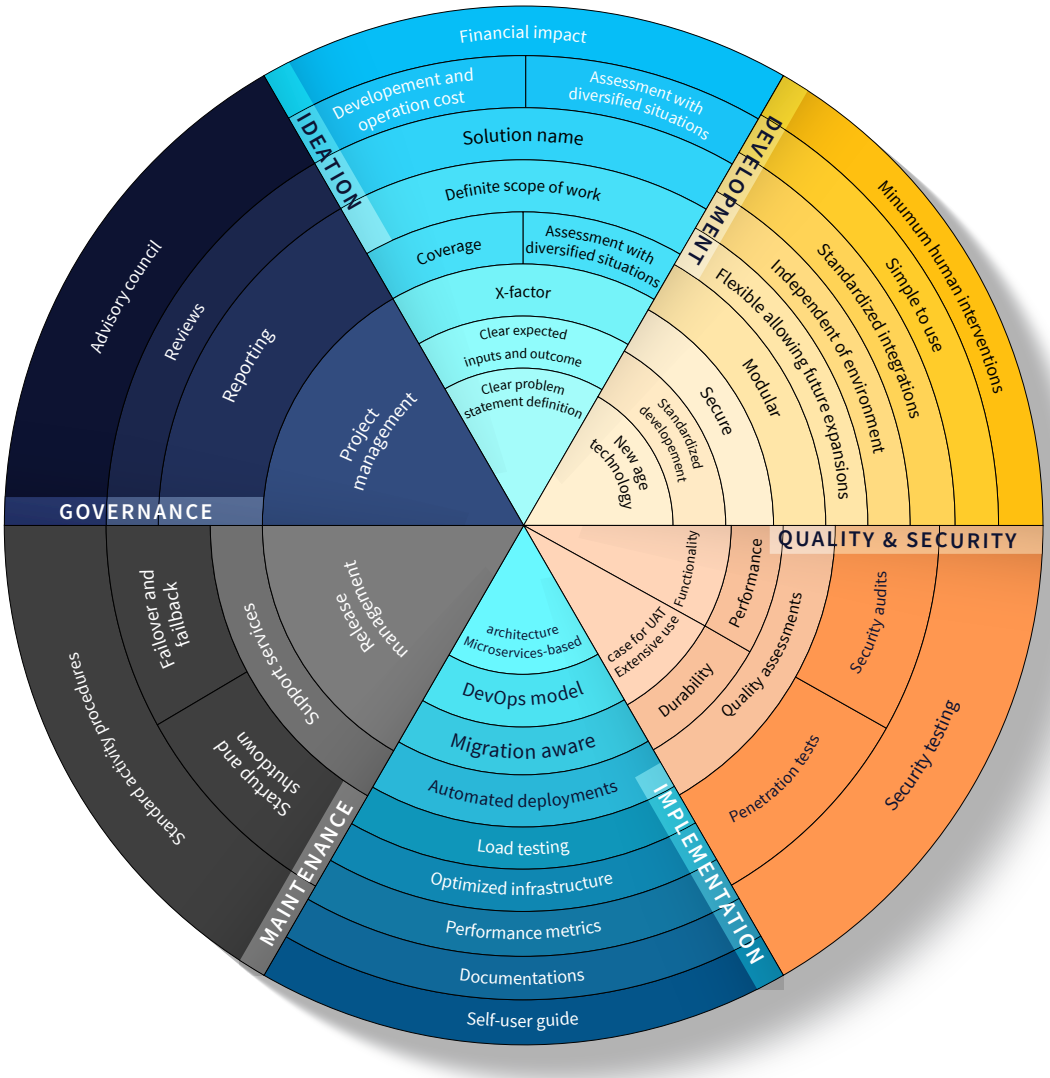

Figure 1: Sustainable automation solution best practices wagon wheel model

Ideation

This is the first stage of the solution life cycle for the incubation of a great idea that can transform into a practical reality. The solution could be a point solution to solve a specific problem or a product that can address collective problems. Irrespective of the nature of the solution, it is essential that solutions are built while following best practices to ensure they are sustainable and deliver real value.

- **Clear problem statement definition:** The first step is to clearly define the problem statement, as this serves as the foundation for deriving expected outcomes and benefits the solution would deliver.
- **Expected inputs and outcome:** The inputs must be well-defined and clear expected outputs must be specified as the resolution of the problem statement. Specifying clear outputs enables us to measure success criteria.
- **X factor:** The solution must have uniqueness with distinguished features that differentiate it from parallel solutions in the market. Users have the liberty to choose from the solutions available. Unless a solution brings an improved or new experience to the user, the idea would be easily dropped.
- **Definite solution scope:** The scope of the solution must have a strong hardline definition to ensure that there are no ambiguities during the development phase. There could be multiple solutions to a said problem, however, we should prioritize and choose a solution or list of solutions that should be added as part of the next phase of solution build. Scope creep will prolong the delivery deadline and bring unrest to the customer experience.
- **Coverage:** It is essential to define the solution's coverage and scope so it can be tested within specific environments or targets. If a solution is tested on a specific target, it gives the user confidence with high output accuracy that would be consumed or experienced by the end user.
- **Assessment with diversified situations:** The solution must be assessed within scope and coverage within diversified situations. This would increase the likelihood of attaining the defined target of the expected result.
- **Solution name:** We need to follow the naming standards and best practices to give the solution an appropriate name. It should fit the brand value and deliver uniqueness that is memorable, understandable, and relevant.
- **Financial impact:** It has the highest impact on the deciding factor for approving the overall idea and sponsoring budget allocation for moving ahead with the next phase of the solution's software development.
    - **Development and operating cost:** This is a cost incurred to give shape to an idea and bring it to life. It is important to perform thorough evaluations of the objective behind the solution and potential benefits that would be achieved by the implementation of the solution.
    - **Return on investment (ROI):** No solution is worth it if it is not going to deliver value and cater to the required objectives. The equation is simple to articulate desired positive outcome. It is the ratio of returns (in most cases it is monetary benefit) to the investment.

Development

After an idea passes the ideation process, it qualifies to go forward in the software solution developement phase. This phase is critically important as it defines the solution's usability, alterations and makes it flexible for sustainable success.

- New age technology: Organizations must select the latest technologies for development purposes to remain at the cutting edge. This would also facilitate easy upgrades, delivering a solution that is bug-free and secure. There are several technologies available in the market including freeware and licensed. It is imperative to decide whether the solution would be hosted on the cloud i.e. IaaS, PaaS, SaaS or on-premise.
- Standardized development: It is a must to follow the standard development process including standard coding, audit, review, and validation processes. Ideally, this should be automated without human interventions in order to ensure there are no biases and make it reusable.
- Secure: A software solution would never get deployed into customer's environment if it is not protected from vulnerabilities and made secure internally and externally for the application's communications. All security best practices should be followed to secure applications from bugs and external attacks to avoid any risks in future.
- Modular: Ensuring modularity makes a solution flexible for users to have custom menus to choose from. It is also essential to keep infrastructure resources proportionate with respect to the modules to justify its modularity.
- Flexible allowing future expansions: Solutions must have flexibility for reusability and amendments to cater to future expansions as requirements would keep changing based on user experience and demand due to the latest trends.
- Independent of environment: Development should be free from hard coding with respect to specific environments. It should be variable-driven with flexibility to expand or shrink based on the requirements of the environment of the deployment.
- Standardized integrations: In case the software solution needs to have integrations with other tools, then it must use the standardized plugins that are simple like a plug-and-play device. The interface should have an easy add/remove feature for installing and removing plugins from the console rather than back-end options. Moreover, it should have role-based access to avoid any accidental disruptions.
- Simple to use: A user-friendly interface plays a crucial role in making any product a success or failure as a user could leave quickly if it is hard to follow and navigate. Appropriate use of wizards, pallets, menus, selection options, etc. make it easy for the user to use the tool/product.
- Minimize human interventions: If a solution is meant to deliver automation, then it should be developed in such a way that minimum or zero human interventions would be needed to execute actions. Human touch at multiple stages would make it a less popular choice for users to adopt if it is supposed to deliver automation.

Quality and security assurance

The solution must pass through rigorous testing for performance, desired functionality, security and durability to ensure it meets the defined standards. It is essential that it meets quality standards.

Quality assurance:

- **Performance:** A solution must pass through the most rigorous testing keeping in mind the solution's performance in varied environments and situations. All possible use cases must be tested to validate the solution's performance. It is essential to create performance benchmarking with respect to every individual component and connector to define the base requirement of the resourcing.
- **Functionality:** All test use cases from user interface (UI) and technical functioning perspective must be validated for desired behaviors of the solution with respect to the features and technical actions.
- **Durability:** Stability and resiliency of the solution becomes a critical factor when a solution is deployed in the production environment. Hence, the solution must be protected and validated from a high availability perspective. For mission critical solutions, it must be protected considering a disaster recovery perspective to make it highly robust.
- Extensive use case for UAT: User acceptance testing (UAT) is a critical phase to ensure we test functionality of the solution to validate the expected outcomes and behaviors.

Security testing:

- **Penetration tests:** It is an essential phase of solution testing from a security perspective. It simulates attempts to exploit weaknesses and vulnerabilities of the solution through network access to it. The solution must be secured against any unauthorized access which could harm the organization's operations and credibility while exploiting security weaknesses.
- **Security audits:** The solution must be assessed and audited keeping in mind security aspects. Auditing should be done internally and externally. It is advised to have an automated mechanism (tool) to audit and generate reports for validations as manual testing could have human errors and biased evaluations.

Implementation

Considering how the solution will be implemented is an essential aspect of the overall life cycle of solution development. In this stage, the decision is made on how the solution needs to be packaged before deployment in the required environment. This phase makes a solution flexible or complex as far as deployment of the solution is concerned.

- **Microservices-based architecture:** Microservices-based architecture enables simplicity of components as they operate independently. Thus, deployment (first-time and consecutive upgrades) and maintenance will become easy in the future.
- **DevOps model:** The modern software's continuous development, integration and deployment mechanism is achieved through the DevOps model. This model makes the overall process faster and robust with minimum or zero impact on operations.

- Migration aware: The architecture should have flexibility and provisions to ensure that any migration with respect to network, domain, environment, etc. would be easy with minimum operational impact for the solution in the future.
- Automated deployments: Customers expect rapid delivery with swift deployment of the solution. Hence, it is imperative that overall deployment is packaged, and the process is automated to eliminate human errors and reduce turnaround time for deployment in the production environment.
- Load testing: The system must be tested with full load to validate and ensure that it would function and deliver during the best and the worst scenarios. Also, it helps to define thresholds for an optimized level of operations.
- Optimized infrastructure: Based on the requirement of the use case of the solution, infrastructure requirements should be optimized to justify its cost. The software solution should have proportionate sizes of infrastructure based on the use cases.
- Performance metrics: All required parameters should be defined and tested to evaluate and validate the solution's performance. These should be added within the solution for monitoring components throughout their life span.
- Documentation: It is mandatory to document all aspects of the solution including technical architecture, functionality, implementation, administration, troubleshooting, etc. It should be amended periodically to make it rich so that it covers all areas of the above aspects.
- Self-help user guide: The success of a solution depends on its ease of usability by the end user. Hence, well-documented user and interactive guides should be made available for users to refer to for easy adaptability of the product.

Maintenance

Once the software solution in developed and ready with deployment essentials, it is critical to make the solution ready for support and maintainance after it is implemented in the respective operating environments. The practices followed post go-lives would make a solution sustainable and ensure user faith in its adaptability.

- Release management: Every software needs regular fixes and upgrades to cater to future requirements and maintain its stability. A well-defined process, procedure and function should be set up to manage all releases (major, minor, bug fixes) of the software.
- Support services: After the software is deployed in a customer's environment and it is serving users, it should operate as specified during design and development. If there are any issues or any incidents raised that make it unavailable for users, support services must resolve the issue. Contact facilities should be set up so customers have seamless access to expert support services.
- **Standard activity procedures**
  - Failover and failback procedure: Mission critical services offered by the solution are always configured with a high availability feature. Hence, the solution should be configured with easy auto failover and failback configurations. And in case services need to be moved manually due to maintenance or performance issues, failover and failback procedures should be documented for support engineers to follow.
  - Startup and shutdown procedure: As part of maintenance activities, it is essential that services are brought down clean and similarly it should be started up appropriately.

Governance

A program controlling mechanism is a critical part of the solution life cycle. Governance plays a vital role to ensure there is a strong focus on achieving defined milestones regularly at every stage. It ensures that stakeholders are updated on progress and timely help is sought to clear any roadblocks with the help of the appropriate team.

- **Project management:** The most popular style of project management is agile as far as product management is concerned. Some common methodologies used alongside agile are Scrum, Kanban and Scrumban.
- **Reporting:** Progress tracking and detailed individual tasks and actions should be translated into the daily, weekly and monthly reporting such as project progress status dashboards, task tracking, milestones, risk and issues tracker, etc.
- **Reviews:** For the successful execution of the project, it is essential to have regular reviews on progress and tracking tasks. Also, the project should be evaluated based on the KPIs defined to gauge the effectiveness of the project management.
- **Advisory council:** A diversified committee should be made up of members from roles such as end user, development team, marketing and branding, leadership, etc. It would be responsible for ensuring that regular feedback (positive and/or negative) is sought from users and other stakeholders and suggestions and recommendations are passed on to the solution development team for further improvements and enhancements.

Conclusion

An idea to resolve an issue or issues is the spark that initiates the journey to developing a software solution. However, sustaining success for a solution in the long-term while fulfilling resolution of business challenges hinges on following best practices. This also serves to cater to all stakeholders' expectations ensuring that the automation solution is successful in all requisite aspects ranging from technical to business perspectives. Hence, it is advisable to have a thorough deliberation to put in place required parameters to ensure the best-fit solution that will make a bigger and sustainable impact.

Author Profile

Navin Singh

Principal – Consultant, LTIMindtree

Navin heads the AIOps platform industrialization and support function. He has 22+ years of experience in IT operation services and automation with specialization in IT service delivery. He is a data science certified professional. At LTIMindtree, he supports deployment of Artificial Intelligence (AI) platforms and products for global customers to embark upon their AIOps journey, enabling them with cutting-edge, tool-based automation.

LTIMindtree

About Us

The world is evolving rapidly with transformative technological advancements, dynamic changes in economies, and a shifting global landscape. These changes make it challenging for our people, clients, partners, and communities to navigate the evolving landscape. At LTIMindtree, we constantly push the boundaries of what's possible by leveraging our expertise, experience, and innovative ecosystem to empower enterprises, people, and communities to build a better Future, Faster. Together. To achieve this, we drive business transformation using what we are good at—technology, talent, and a robust ecosystem of partners—to eliminate all barriers to progress. Our commitment is to a singular goal: to relentlessly ensure our clients become their future sustainable selves ahead of schedule.

LTIMindtree is a global technology consulting and digital solutions company that enables enterprises across industries to reimagine business models, accelerate innovation, and maximize growth by harnessing digital technologies. As a digital transformation partner to more than 700 clients, LTIMindtree brings extensive domain and technology expertise to help drive superior competitive differentiation, customer experiences, and business outcomes in a converging world. Powered by 81,000+ talented and entrepreneurial professionals across more than 30 countries, LTIMindtree — a Larsen & Toubro Group company — solves the most complex business challenges and delivers transformation at scale. For more information, please visit https://www.ltimindtree.com/.