LTI

Let's Solve

Agility Architecture for **Insurers**

A Larsen & Toubro Group Company

Agility Architecture for Insurers

The insurance world is getting disrupted. The Insurer's traditional role of a 'Risk Carrier' is shifting to more of a 'Trusted advisor', and they are increasingly taking control of end-to-end customer experience. The increased competition is forcing them to innovate and bring in newer products to the market very frequently. As sensor technology is opening up opportunities for real-time risk monitoring and loss prevention, Insurers are forming a partner ecosystem to provide value-added services to customers.

Technology plays an important role in this new world. Insurers need 'agility' in their architecture to constantly innovate and improve customer experience. In their current world, a new product release takes as long as 12-18 months after conceptualization, while a new customer interaction feature takes at the least 3 months. With the speed of change, this is no longer acceptable. They need to be able to get to Facebook-like daily releases. Along with agility, they also need more resiliency, as they cannot afford to be 'not-available' in a constantly connected world.

Most insurers have adopted agile practices to improve their IT delivery; however, it is not enough to merely follow the practices. They also need infrastructure and architecture that supports 'agility' and 'resiliency'.

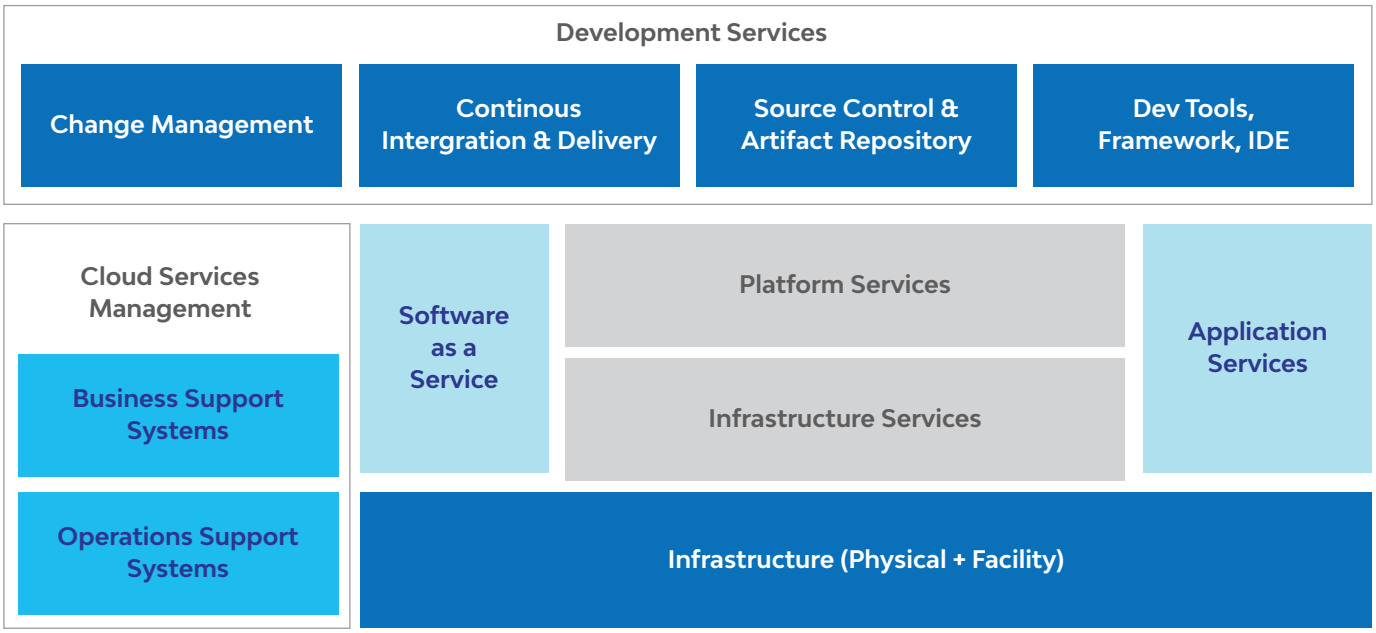
In this PoV, we have described the new agility architecture, mostly based on Open Source stack, that insurers can leverage to improve agility and resiliency.

What is Agility Architecture?

Agility Architecture is an effort to keep the architecture solution simple, and achieve maximum automation for repeatable task around solution changes and release. This, in turn, enables the business to be agile and adapt to the changes in time of needs.

Agility Architecture building blocks involves -

- Agile Platform
- Agile Development Services
- Application Patterns & Principles
- Business & Operations Support Systems meeting objectives of an agile organization



Agile Platform

When we talk about Agile Platform, "Cloud" is considered the cornerstone. As a first step towards adopting cloud, organizations took two major steps-

- Step 1 - Virtualizing their physical infrastructure. This allowed organizations to dynamically create new virtualized environment without having to wait, while procurement of new systems work out. Infrastructure as a Service (IaaS) was leveraged to manage and control the underlying infrastructure and provision of virtualized cloud infrastructure like network, servers, operating systems & storage.

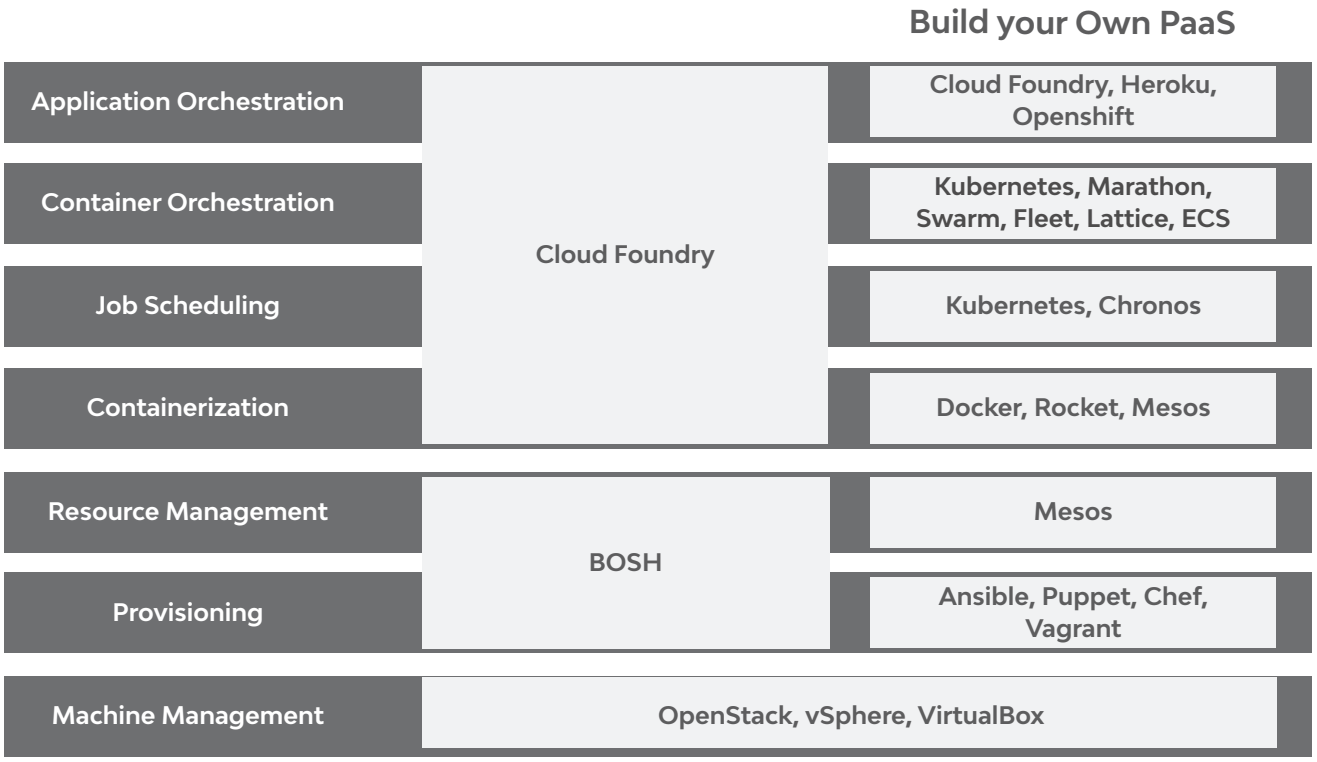
- Step 2 - Adopting packaged software via Software as a Service (SaaS) model. SaaS uses the web to deliver applications that are managed by a third-party vendor, and whose interface is accessed on the clients' side.

For an enterprise, which built their own application, they needed a platform that allows customers to develop, run, and manage applications. This is to be done without the complexity of building and maintaining the infrastructure, typically associated with developing and launching.

Platform as a Service (PaaS) emerged as a capability that allowed organizations to bring abstraction over the compute resources and create their private cloud ecosystem. The platform also allowed deployment and orchestration of application created using supported programming languages, libraries, services and tools.

All major product companies have come up with a PaaS product, for e.g. AWS Elastic beanstalk, Google App Engine, Heroku, IBM Bluemix, Pivotal, etc. With the growing open source ecosystem of cloud related framework and tools, you can actually building your own PaaS.

Given below is a quick representation of all the major building blocks of PaaS. In the open source ecosystem, you can either choose a PaaS product. For e.g. Cloud Foundry, or pick and choose the best of the framework and create a connected PaaS as shown below -



Any platform is only powerful as its ecosystem of the application services. Application services are necessary for rapid application development, and are cloud-native. Given below are some of the common services that are being leveraged by the enterprise application.

Platform as a Services have gone beyond the basic application services ecosystem, and have added additional platform services for mobile and Internet of Things. Example of one such open source platform is Cloud Foundry. Finally, security is one of the major considerations of any agile platform. This includes following key aspects:

- Authentication - A way to authentication by augmenting username/password credentials, with a hardware or software RSA/-JSON-based token.
- Identity management - An effective identify management solution to manage the consumer usernames and/or integrate to an in-house system such as Microsoft Active Directory.
- Security monitoring - Tools to track and identify any potential security issue.

Agile Development Services

For an enterprise to effectively deliver its business solutions in an agile manner, development services need to have key components -

- Continuous integration and delivery
- Version control and artifact repository
- Development tools, framework & IDE
- Change management

Agility Architecture for Insurers

Applications need to adopt new patterns and principles that will be able to leverage the capabilities of the agile platform. These cloud-native design or architecture were designed, considering following principles in place:

- Use declarative formats for setup automation, to minimize time and cost for new developers joining the project
- Have a clean contract with the underlying operating system, offering maximum portability between execution environments
- Are suitable for deployment on modern cloud platforms, obviating the need for servers and systems administration
- Minimize divergence between development and production, enabling continuous deployment for maximum agility

The principles described above were formulated by Heroku in form of "12 factor app". The principles discussed in 12 factor apps are given below:

Codebase
One codebase tracked in revision control, many deploys

Dependencies
Explicitly declare and isolate dependencies

Config
Store configuration in the environment

Backing Services Config
Treat backing services as attached resources

Build, release, run
Strictly separate build and run stages

Stateless process
Execute the app as one or more stateless processes

Port binding
Export services via port binding

Concurrency
Scale out via the process model

Disposability
Maximize robustness with fast startup and graceful shutdown

Dev / Prod parity
Keep development, staging, and production as similar as possible

Logs
Treat logs as event streams

Admin processes
Run admin/management tasks as one-off processes

Containers, Microservices & Serverless Architecture

Containers are portable, effectively running on any hardware that runs the relevant operating system. That means developers can run a container on a workstation, create an app in that container, save it in a container image, and then deploy the app on any virtual or physical server-running the same operating system - and expect the application to work.

On the other hand, Serverless A capital architecture is evolving, but support for all development frameworks may not be available. One of the main consideration is vendor lock-in - such lack of control may manifest as system downtime, unexpected limits, cost changes, loss of functionality, forced API upgrades, and more.

Adoption Considerations –

- What tool should we consider for orchestration?
- Type of containers – LxC, Docker, Mesos
- Support for current application technology – ease for Lift & Shift
- Image repository selection
- Application business layer decomposition
- Front end SPA decomposition
- Cloud vendor Azure/AWS/Google
- Moving to .NET Core & available web server support in cross-platform
- Selection of type of serverless service Cloud provider – Azure Functions, AWS Lambda
- API Gateway tools consideration in hybrid conditions

Business & Operations Support Systems

As the cloud ecosystem grows, it is necessary for the organization to effectively manage and service it. There are two aspects pf enterprise services that need to enhanced or modified -

- Business support systems - These processes are business-oriented and focus on the business operations of your private agile platform. Cloud computing applications, data and IT resources are presented to users through self-service portals. Some of the key functions are -
  - **Customer Management -** This area covers the activities necessary to manage and maintain the relationship with the cloud consumer. It deals with items such as customer accounts, complaints and issues, customer contact information, history of customer interactions, etc.
  - **Contract Management –** It establishes service level agreements for cloud services being delivered to customer.
  - **Service Catalogue –** It establishes a service catalog and is the primary interface for the consumer to engage with the cloud provider.
  - **Billing & Metering –** It establishes a way to account for cloud service usage and implement chargeback invoicing.
  - **Reporting & Auditing -** This function monitors, tracks, and logs activities performed by the consumer, usually through the management console.
  - **Pricing and Rating -** This process establishes the price points and Tiering for the cloud services

- Operations support systems -These process that are focused more on the infrastructure management methodology for managing large numbers of highly virtualized resources. These virtualized resources can reside in multiple locations, but still perform as a single large resource to deliver services. Some of the key function are -
  - **Provisioning and Configuration –** These functions deal with automated self-service provisioning of the infrastructure and automated upgrade/replacement of capacity or components.
  - **Monitoring and Reporting -** Ongoing monitoring of the operations and cloud infrastructure is critical to ensure effective and optimal quality of service.
  - **Portability and Interoperability -** Ensure that enterprises are not caught up in vendor lock-in situation. This function should provide seamless ability for data portability, service interoperability & system portability.

info@Lntinfotech.com
www.Lntinfotech.com