WHITEPAPER

Vanenburg's vision on modern enterprise architecture

vanenburg

Content

Introduction 03

Pace-layered approach 05

MASA approach 06

Composable Enterprise 08

Vanenburg's vision 10

Why Vanenburg 17

vanenburg

Introduction

In an era where organizations require a high level of flexibility to stay aligned with the market needs and benefit from new business models and technologies, the IT infrastructure is more and more measured on its agile capabilities and business outcomes. Evolving digital business demands, partnerships, and user expectations requires an application architecture that supports agility, flexibility, and innovation, enabling the business to jump into new opportunities quickly.

Organizations are increasingly moving away from a situation where business and IT are seen as completely separate domains in which they have to work with the possibilities dictated by IT.

Traditional n-tier application architectures are slowly being replaced by multi-grained, cloud-native enterprise architectures, such as Mesh App and Service Architecture (MASA) and Microservices Architecture (MSA) that support multiple channels: web apps, mobile apps, social media, notifications, email, and more.

Competitive pressures drive organizations to provide user experiences that exceed expectations, and leading digital business players ensure that users have optimized, continuous and ambient experiences as they traverse these channels.

These changes set the stage for others. Organizations will increasingly fall behind if they aren't prepared to deliver these multi-channel experiences, leverage third-party algorithms and data sources, or rapidly provide new features.

However, this journey from a traditional n-tier application to modern architecture can be complex. Many paths can take you from your current state to modern architecture.

Typically, these paths consist of a combination of the following aspects:

- Move infrastructure to the cloud: This will allow the organization to get rid of the technical hassles related to the maintenance of server hardware and focus on the added value to be delivered by the application landscape. The flexible, pay-per-use infrastructure makes it possible to eliminate the need to pre-investment in hardware and frees up financial resources to invest in measures to modernize the application architecture.
- Decouple components and layers in the application architecture where it makes sense: Decoupling can bring additional overhead, so this should be done based on careful considerations. In recent years, Gartner developed several methodologies to guide organizations in making the right decisions in this regard, such as:
  - *Pace-layered Approach*, a methodology to govern software applications through their entire life-cycle in support of evolving business requirements.
  - *MASA*, a Mesh Architecture of apps, APIs, and services that aim to serve experience through multiple channels.
  - *Composable Enterprise*, a design approach that embraces the API economy, delivering business outcomes through the assembly and combination of Packaged Business Capabilities.

Let us walk through these concepts and see how they each address the aspect of decoupling.

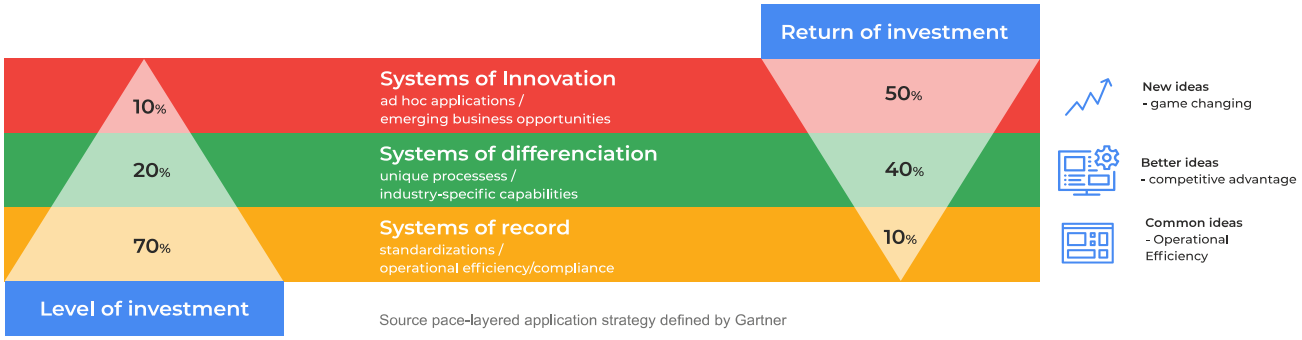
vanenburg

Pace-layered approach

Gartner's Pace-layered Application Strategy is a methodology for categorizing, selecting, managing, and governing applications to support business change, differentiation, and innovation. It categorizes applications by process or function and asks whether the application module supports a common requirement, a unique business methodology, or an innovative new business process. This allows the organization to apply the appropriate governance, funding, and data models, based on the characteristics of each application.

Pace-Layered strategy and the road to innovation

The speed of change and the impact on various enterprise IT Layers:



Source pace-layered application strategy defined by Gartner

Having these layers in place and establishing common elements as connective tissue between them allows the organization to become more agile to change and provides in a controlled manner the ability to experiment with logic that potentially will increase competitiveness. Governance in the upper layer will typically give more room for experimentation, alternate platforms, and agile practices. It is much more practical for developing features such as for instance a mobile application, a recommendations engine or a sentiment analysis service.

This methodology has contributed significantly to the industry conversations between IT and the Lines of Business (LOBs). Others have worked this out to a more detailed level. Salesforce introduced *"Fabrics, a reference model for the journey to the cloud"* as a guide to increase the understanding between Enterprise Architects and LOB stakeholders by dividing *"slow IT"* and *"fast IT"* in the process of setting joint priorities.

MASA approach

While the Pace-layered Application strategy applies decoupling primarily from the angle of business agility, the MASA[1] architecture looks at this more from a *reuse perspective*. The agile architecture of MASA comprises multiple independent and purposeful modules that can be composed as needed to support multiple channels, users, roles, and networks for continuous delivery of application functionality and new capabilities. This architecture supports diverse data sources, leverages pertinent context information for better-automated decisions, and exploits the API economy and the external algorithm marketplace. The *distributed nature* of MASA can generate *increased operational* complexity, but it improves agility and scalability, which are primal requirements for any digital business.

Typically a MASA-based application will be designed to be cloud-native, making it ultimately scalable. Its modularity and adaptiveness help to enable dynamic business changes that can be developed one service at a time without cross-dependencies.

MASA contributes to implementing *digital business ecosystems* on a larger scale by offering an optimized and consistent solution across any device. It links everything from mobile apps and desktop apps to web apps and the IoT into a vast mesh of back-end services, culminating in creating applications for user interactivity.

1 Gartner “The Top 10 Strategic Technology Trends for 2017”

MASA requires a proper design to make sure the systems will run reliably and securely. The attack surface of a MASA network is potentially larger compared to that of a monolithic core system, so appropriate controls should be brought in place accordingly. But that this is sufficiently possible is proven by Google that offers various services with more than a billion users in the same very elastic manner. A fascinating read in this context is the book "Building Secure & Reliable Systems".

The MASA concept is an enabler of the following important trends:

- the *"intelligent digital mesh"*, which helps transform business models to accommodate people, processes, and smart devices in a collaborative framework to achieve more value together than individually as lone parts.
- the *"democratization of development"*, which allows citizen developers to develop tailored user experiences with a cohesive and consistent user journey by maximizing the reuse of back-end services.

Composable Enterprise

Another term coined by Gartner is the *Composable Enterprise*[1]. This builds further upon the previous concepts and refers to an organization that delivers business outcomes and adapts to the pace of business change through the assembly and combination of Packaged Business Capabilities (PBCs). PBCs are application building blocks that have been purchased or developed.

This concept assumes that software vendors will modernize their legacy applications towards this PBC model and that this will become the prevailing architecture, integration, and delivery model for software innovation. Organizations need to prepare for this and convert their application landscape towards a plug-and-play architecture. The various PBCs will connect through APIs and Event Channels. This application architecture enables the assembly and dynamic reassembly of PBCs, which can be of the following kind:

- Sourced from various *third parties, including traditional packaged software vendors, PBC marketplaces, and nontraditional parties (such as financial services companies and global service integrators)*.
- Developed *internally* — either by central IT or within business units — providing net new capabilities.
- Delivered through the *modernization* of monolithic applications into more modular components.
- Created or *assembled* by the organization's employees, empowered by low code platforms - increasingly having the technical skills to deliver business value.

This architecture is *designed for change* and enables the creation, curation, and dynamic reassembly of PBCs. It is not a new architecture. Rather, it is an ongoing evolution of architecture practices building further upon the Service Oriented Architecture theme, but differing from that in the sense that a PBC should always be functionally recognizable by a business user. The popular iPaaS platform MuleSoft supports this concept of Composable Enterprise very nicely. They divide their APIs into three separate layers: "System APIs," "Process APIs" and "Experience APIs".

1 Gartner "Future of Applications: Delivering the Composable Enterprise"

In the ERP domain, we see a shift towards enterprise applications complemented by an ecosystem that includes new capabilities for application platforms, integration, and low-code/no-code development. Gartner refers to this new era of ERP as *"Enterprise Business Capabilities"* (EBC). According to Gartner, by 2025, the top four ERP vendors will rebrand themselves as a *business platform provider*.

When we take SAP as an example, we clearly see this happening. SAP is actively pushing its customers to move to the cloud and adopt the modern HANA-based architecture. A shift towards more genericity is required to maximize cloud benefits. Next to the option of *Classic Extensibility* which despite its freedoms can only be used on AnyPremise (onPremise or PrivateCloud) but oftencomplicates the upgrade path, customers have now the choice to implement their custom wishes using two other approaches:

- In-App Key User Extensions: This is the way of performing customization directly on the SAP stack. The customization is done by configuration and limited ABAP coding and will not negatively impact the upgrade path. This can be used to enhance the data model, apply custom business logic that touches the transactional core, and expose selected data to the outside world so that you can further build upon it outside SAP. The benefits are that it enables the key user, is fast and lightweight, and reuses the existing objects and components in SAP.

- Side-by-Side Extensions: This allows 3rd party or custom applications to run integrated with the SAP core instance. SAP has a growing ecosystem of partners offering supplementary applications or low code tooling on top of the main SAP logic. Also, here, the upgrade path is not negatively affected. The extension factory supports integrations on various layers: data replication, processes, events, rules, workflows, and user experience and offers capabilities for integration, development, deployment with multiple runtimes, operation, and provisioning. The benefits are that it allows you to use your technology of choice to develop logic. In addition it supports decoupling certain components to provide them more globally in the application landscape and makes it easier to combine logic with data from other back-ends.

An organization's ERP strategy must utilize the potential of their ERP provider's ecosystem and underlying platform delivered by vendors, enabling rapid adoption of new capabilities.

Vanenburg's vision

So decoupling is key when it comes to making your enterprise software *agile*. It's an important enabler for the *democratization of IT* and the closing of the *business-IT gap*. It opens up the application landscape to embrace the opportunities that come with the era of the *Composable Enterprise*. It will also further increase the benefits of cloud computing by incorporating *serverless computing* and *managed services*. And last but not least, it will give the possibility to incorporate *ML-based solutions*, which is often a source of gaining competitive advantage.

However, where to exactly decouple is not an easy decision. It depends on the capabilities of your current application landscape, the vision and roadmap of your ERP provider and the skills internally available in the organization. Therefore the organization needs to balance the various options carefully.

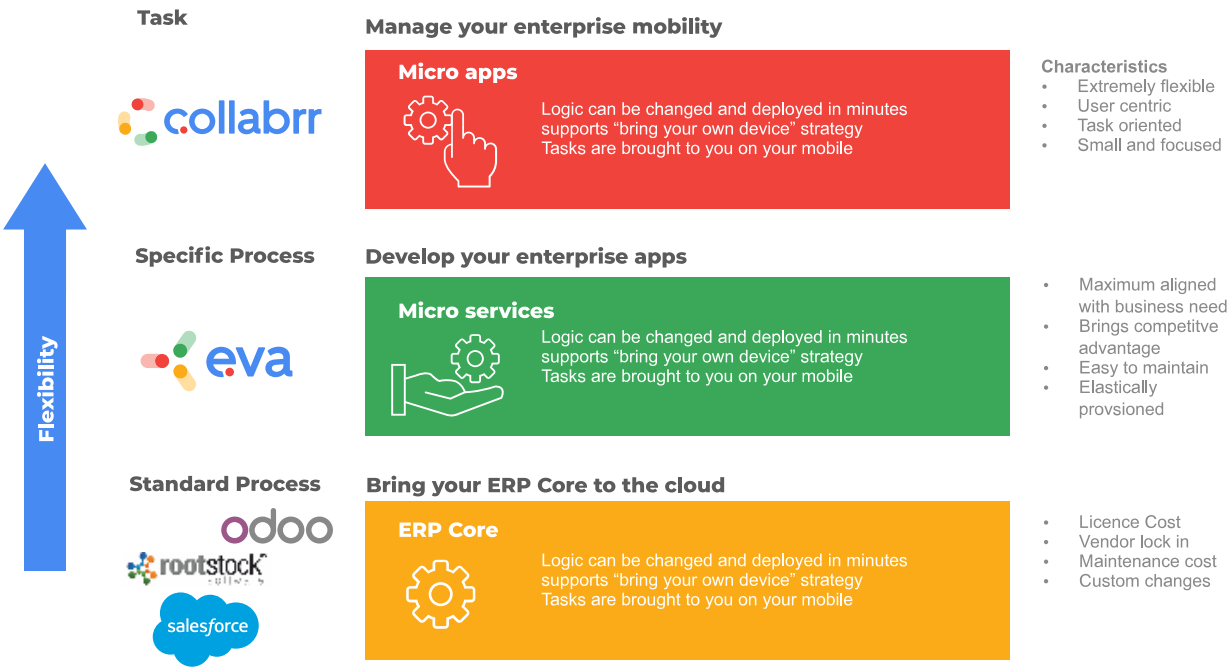
In this consideration, it is important to avoid the following pitfalls:

- Avoid making it *unnecessarily complex*. A monolith can be complex, but a MASA architecture without the correct governance has the potential to become highly complex as well. If that's the case then use service mesh tooling to manage the integrations. Make sure there is a full overview and accountability.

- *Avoid vendor lock-in* or take at least informed risks on this topic. Design a proper exit strategy at the start of the project. If possible, use open source technologies, such as Kubernetes or MongoDB, and design your application to be MultiCloud which practically will include AnyPremise.

- Avoid too many customizations in proprietary software. Instead, soften the pain of using proprietary software by sticking to the standard to be more easily decoupled to governance and will not consume a ton of energy from the organization to sustain.

As Vanenburg, we often use the diagram below to brainstorm with our customers on enterprise architecture modernization.



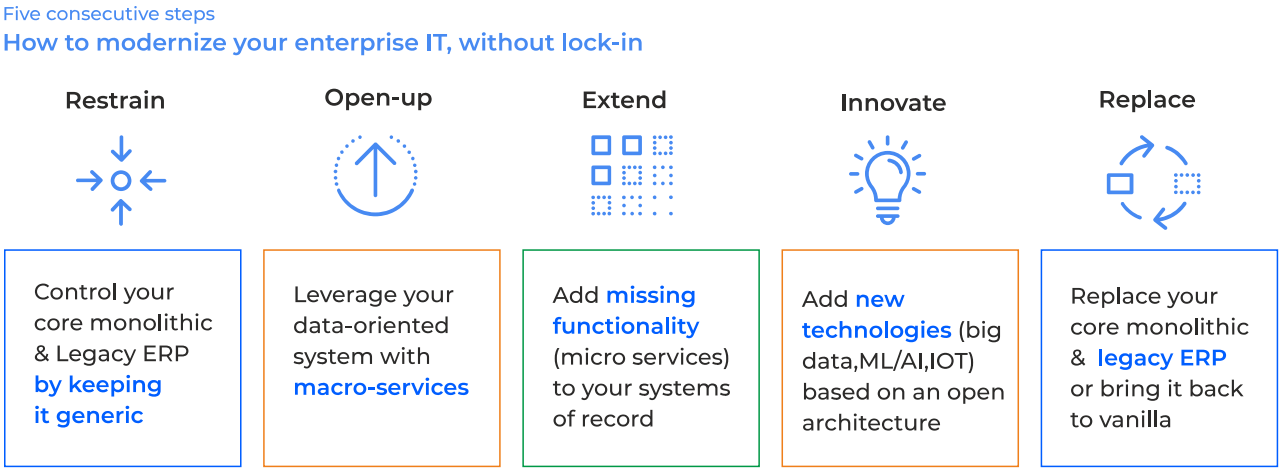
In our conceptual thinking, we differentiate between the following layers:

Layer 1: Core transactional logic, meant to support standard processes. In this layer, for example, off-the-shelf ERP and Accounting solutions have a place. They are designed for compliance and support the broadly accepted best practices for the domain. Unless the transactional processes of the organization are very special, it will generally make less sense to develop custom systems for this layer. Our advice is to minimize custom changes that will negatively impact the upgrade path and carefully look at the scope of the systems to avoid monolithic complexity, with the related high maintenance costs, license costs, and increased vendor lock-in.

Each customer situation is different, and as earlier stated, it depends much on the capabilities of your current application landscape, the vision and roadmap of your ERP provider, and the skills internally available in the organization. Also, the vision and cloud strategy of the customer does have an impact on the steps to advise.

We hear often from our customers that they are satisfied with the base functionality that their core system provides but that they feel a lack of support for the many upcoming innovations that they hear about from others. Core system providers often have a large customer base to serve, sometimes even in different versions of older technology, hindering them from embracing innovations quickly. If that's the case, then we often help our customers to get in touch with their core system provider to have them open up their system so that applications on more modern technologies and frameworks can be developed on top of it in a decoupled manner.

When we see customers struggling with an ERP system that is limiting their flexibility and innovation power but they don't have the luxury to move away from it, then we guide them using the following five consecutive steps:



In other situations, customers cannot find an off-the-shelf solution that covers at least 80% of their needs, and they decide to build the core system themselves. In the past, many homegrown solutions were developed based on Microsoft Access and SQL Server. Often the knowledge was only available to a small team of developers, which posed a risk to the organization.

When the company wants to be deeply involved in the design and development of the ERP system, the Thinkwise platform offers a decent solution. It brings domain experts, citizen developers and core developers together in a collaborative environment and controls the whole lifecycle from requirement gathering to feature delivery and usage.

The Salesforce platform also offers core applications with the ability to customize those or add custom extensions to them. There is an AppExchange where modules from third parties can be purchased and added to the application landscape in an integrated way with the existing core functionality.

For startups and small companies that still have to select their core systems, Odoo can be a great choice. It can be used as SaaS from Odoo itself or deployed on AnyPremise. In the latter case, it is highly customizable. The open-source version can be used free of charge, but also the commercial edition is relatively cheap and provides you with more features and good support.

If your ERP provider is on top of the new developments and is actively working on adjusting and moving its architecture to the new era of Composable Enterprise, and you are satisfied with them supporting your business, it might be the best choice to quickly accept each of their new releases and migrate to the cloud if possible.

SAP, for instance, offers their clients the RISE with SAP program, which helps them simplify the transformation of their business for the digital age.

During the assessment of this core layer, we typically discuss the following questions with our clients:

- What is the maturity level of the organization when it comes to standardized processes? Those core processes are typically not yet stable in a startup, and the needs may fluctuate over time.
- What are the possibilities to modernize the core transactional systems?
- Will the core system provider offer an attractive path towards an open, cloud-based architecture?
- Can we solve the custom needs outside of the core system, or is the functionality too much related to the core to take it out?
- Are there any business reasons to split off certain pieces of functionality, for instance:
  - Outside the core system, they can be easier reused on a global level.
  - Experiment with the functionality and the ability to quickly adapt and try out various things in the market.
  - A highly qualified team that can easily build new functionality in another technology.

Layer 2: Specific process. While in Layer 1, the design-time change responsibility is often outside of the organization, in Layer 2, the organization will be in charge of the *design-time changes*. The custom logic defined by the organization will be developed in separate components and aims to bring the organization a competitive advantage. A maximum alignment is guaranteed by having the business heavily involved in the generation of those components.

During the assessment of this specific process layer, we typically discuss the following questions with our clients:

- What will be the delta need for functionality when the core systems are kept generic and small or modular?
- How can this functionality be split up into logical components?
- From a functional perspective.
- From a technical perspective.
- From an organizational (facilitating teams) perspective.
- Can the required data be exposed from the core systems?
- How do we support the analytical needs of the organization when the data is stored in multiple applications and microservices.
- Do we need technology to support integrating microservices, such as an iPaaS platform or Mesh framework? Some cloud services, such as Google App Engine, have this already built-in.
- What technology will be used to develop the microservices?

If Java is an option for the organization, we showcase the capabilities of our Eva productivity improvement framework which is excellent for quick prototyping together with the business and generation of code afterward, and helps to drastically reduce the lead time and costs for the development of those microservices.

Layer 3: Task, meant to support the daily work of the knowledge worker. While in Layer 2, the organization is in charge of the design time changes, in Layer 3, the knowledge worker is in charge of run time changes. Therefore the tooling needs to be extremely flexible and configurable. It's like a spreadsheet in the hands of a knowledge worker, but then in a connected and controlled manner.

During the assessment of this task layer, we typically discuss the following questions with our clients:

- What is the tooling currently used by the knowledge worker to support their day-to-day work, especially what is developed and used outside of the governed systems? For example, in many organizations, sophisticated excel based logic is used in an uncontrolled manner but of vital importance.
- Does the organization have skilled power users that would be able to make quick adjustments to the logic when required?

We showcase the capabilities of our platform Collabrr which offers organization-controlled provisioning of task-oriented apps to the organizations and its business network integrated with the data and logic in the other layers.

Although this has proven to be an effective way to analyze an organization's options for architecture modernization, the disclaimer is that the boundaries between those layers are not always easy to define and which layer to position a system is not always black-and-white. For example, a platform like Salesforce provides standard processes (Layer 1) but also out-of-box support for customizations with the flow builder (Layer 2) and also support for the knowledge worker with customizable dashboards and tasks (Layer 3). So there might be good reasons to use the platform for all layers.

In the end, technology is not a goal in itself. Instead, an important thing that makes technology good or bad is whether it supports my business and makes me ready to embrace new business opportunities. Now and in the future.

Why Vanenburg

- 40+ years of experience in the Enterprise Software market with deep expertise in ERP and integrations to back-end systems.
- Dedicated Google Cloud Practice since 2014 with +50 FTE.
- Unique 'Rapid Application Development' tooling to quickly prototype and develop applications at large scale against low total costs of ownership.
- Fully automated cloud hosting and management service based on:
  - "Infrastructure-as-Code" approach to provision and manage (24*7) cloud resources and applications;
  - Strong artificial intelligence capabilities based on Google AI.
- Interested? Contact us at info@vanenburg.com to learn more or schedule a demo.

Selected customers:

Vanenburg is an independent IT service provider. We are experts in developing Enterprise IT Modernization solutions combined with the services we offer on the Google Cloud Platform, Salesforce Platform, and open Java technologies.

Jan Baan founded Vanenburg in 2009 as part of the Vanenburg Group and its core team has 40+ years of experience in the enterprise software market (Enterprise Resource Planning, Business Process Management and Smart Process Apps) – building upon a history of successful IT innovations like Baan ERP and Cordys Cloud BPM.

For more information, please visit www.vanenburg.com. Follow us: