

# TOWARDS SERVERLESS ENTERPRISES

Developing the Enterprise Serverless Assessment (ESA) to assess and improve an organization's fit and readiness for Serverless technology

Thom T. J. Leemans

April 2022



# TOWARDS SERVERLESS ENTERPRISES

Developing the Enterprise Serverless Assessment (ESA) to assess and improve an organization's fit and readiness for Serverless technology

A thesis submitted to the University of Twente in partial fulfilment of the requirements for the degree of

**Master of Science in Business Information Technology**

**Commissioned by**

Capgemini Invent

**University Supervisors**

Dr. Lucas O. Meertens

Dr. Faiza A. Bukhsh

**External Supervisors**

Daniël Visser, MSc.

Christiaan Tick, MSc.

**Author**

Thom. T. J. Leemans

**April 2022**



# Executive Summary



Serverless is the next step in cloud computing that promises a faster time-to-market, improved agility and elasticity, higher availability, a lower total cost of ownership, and more efficient use of human resources. This innovation allows organizations to outsource all server management to a cloud service provider enabling developers to focus on what they do best: writing code and developing the core business logic. This enhanced developer productivity lowers the cost of development and operations while mitigating operational risks. The service provider ensures that the application's infrastructure is always available when needed and can infinitely scale them within microseconds to ensure they are rigid and cost-effective. You only pay for the resources you use.

Research indicates that most enterprises want to start using Serverless in the upcoming two years, but their adoption is lagging due to a lack of guidance. This lack provides an opportunity for consultancy firms to serve their clients and led to the discovery of a research gap among available Serverless frameworks. Each of the available frameworks skips the initial requirements analysis phase. At the same time, other sources state that a preliminary analysis of the fit between an organization and Serverless is crucial for its success. Therefore, we aimed to solve this research problem by studying what high-level requirements organizations can meet using Serverless and when they are ready to adopt it. This goal led to the Enterprise Serverless Assessment (ESA), which fills the research gap for enterprise organizations and delivers the insights they need to kickstart their adoption. The ESA provides consultants with a solid base to determine an organization's strategic fit and readiness to adopt Serverless, enabling them to perform the missing requirements analysis phase at an overarching level. After the assessment, the organization knows if Serverless aligns with its strategy and if they have the required capabilities for a smooth and successful adoption. By analyzing the organization from multiple perspectives, consultants can conclude the assessment with advisory reports on how the assessed organizations can improve their readiness.

By gathering an expert panel of Chief Technology Officers, architects, and developers, we criticized the theoretical knowledge available in the literature and complemented it with practical knowledge. During multiple rounds, the experts openly contributed their experiences and views, after which they validated the input of others to ensure every addition was well-substantiated. This approach resulted in a comprehensive list of criteria divided into seven assessment topics proven to develop robust cloud platform strategies. We structured all findings within a tool that enables every consultant to carry out the ESA effortlessly.

We assessed a multinational financial services corporation that is advanced in its Serverless adoption already. This execution demonstrated the usability of the ESA, and their experience allowed direct evaluation of the results. The findings aligned with their self-perception and would have supported their adoption at the beginning of the process.

To conclude, this study fills the research gap among existing Serverless frameworks and enables consulting firms to capitalize on the growing enterprise Serverless market. Through this study, we advanced the scientific state of Serverless by solving the research problem and by criticizing, substantiating, and complementing the knowledge available in both grey and white literature. This process allows scientists to use the advanced grey literature in the future and led to (partly) solving multiple other open research issues concerning Serverless in the process.



# Acknowledgements



This thesis marks the end of my study and student life. Although this time of my life has gone entirely different than I expected beforehand, I can confidently say that I am satisfied with how it turned out. I am grateful to all the people who participated and supported me. Since the process was a team effort, I would like to take this opportunity and acknowledge a few people.

During this research, my supervisors from the University of Twente were Lucas Meertens and Faiza Bukhsh. They gave me a lot of autonomy to challenge myself. When I had questions or faced setbacks, they were readily available to answer them and get me back on track. They were flexible and constructive, which enabled me to get the most out of this research. I am thankful for all their guidance and the time and effort they spent to make this research a success.

I was fortunate enough to perform this research on behalf of Capgemini. This organization allowed me to invent my own research assignment while accommodating guidance and support. Especially my direct supervisor Daniël Visser put tremendous effort and time into formulating the research with me, gathering experts and enterprises to participate, and assisting me during all phases of the thesis. He ensured I felt welcome in this organization and taught me how to use the organization's strengths to achieve my goals. Christiaan Tick was my second internal supervisor, who was always available to provide input and feedback to further the research and my capabilities. We often talked about my ambitions in life, and he helped me understand what drivers to look for in work.

In addition, I wish to acknowledge all the experts that participated in the expert panel and during the demonstration. Their input has been vital and enabled me to conduct the research interactively. It allowed us to criticize existing literature and complement it with new practical knowledge.

Finally, I would like to thank my family and friends for their continuous support and making my student life a real treat. When times were tough, you were always there to encourage and motivate me. I am looking forward to what we will experience together in the future.

This thesis was the perfect ending to my study and prepared me to start my professional career. I hope you enjoy reading it and that it will lead to new insights.

Thom Leemans

*Utrecht, April 2022*





# Table of Contents



1.	Introduction: Identifying the problem and formulating the research plan .....	1
1.1	Context .....	1
1.2	Opportunity Identification .....	1
1.3	Research Gap Identification .....	4
1.4	Research Approach .....	6
1.5	Research Design .....	8
1.6	Research Structure .....	11
2.	Theoretical Framework: Gathering the theoretical knowledge needed to start the development... ..	13
2.1	Definition .....	13
2.2	Promise .....	16
2.3	Suitability .....	20
2.4	Technology .....	22
2.5	Summary .....	26
3.	Design & Development: Realizing the Enterprise Serverless Assessment (ESA) .....	29
3.1	Assessment Topics .....	29
3.2	Literary Criteria .....	31
3.3	Expert Criteria .....	33
3.4	Prioritization .....	41
3.5	Assessment Tool .....	42
4.	Demonstration: Performing the ESA within conditions of practice .....	47
4.1	Questionnaire .....	47
4.2	Results .....	48
4.3	Advice .....	49
5.	Evaluation: Determining the effectiveness of the ESA in guiding organizations .....	53
5.1	Usability & Effectivity .....	53
5.2	Limitations .....	56
6.	Conclusion & Recommendations: Completing the research and defining what is next.....	59
6.1	Conclusion .....	59
6.2	Recommendations .....	62
6.3	Future Research .....	62
	Bibliography .....	65
	Appendix 1: Delphi Details.....	69
	Appendix 2: Delphi Consensus.....	73
	Appendix 3: Prioritization Details.....	75
	Appendix 4: ESA Tool.....	77
	Appendix 5: Demonstration Details .....	83



# List of Tables



- Table 1: Cost comparison between EC2 and Serverless ..... 18
- Table 2: FaaS and PaaS costs comparison ..... 21
- Table 3: Provider cost comparison ..... 24
- Table 4: Action Prioritization Scores ..... 41
- Table 5: Architecture Framework questionnaire part during demonstration ..... 47
- Table 6: Demonstration Results ..... 48
- Table 7: Delphi Details - Round 1 ..... 70
- Table 8: Delphi Details - Round 2 ..... 71
- Table 9: Impact/effort rating translation ..... 75
- Table 10: Prioritization details ..... 76
- Table 11: Demonstration Questionnaire Details ..... 85



# List of Figures



- Figure 1: The research cause ..... 1
- Figure 2: The thirteen Enterprise Goals..... 2
- Figure 3: Serverless publications over the years..... 2
- Figure 4: Mapping enterprise goals onto Serverless characteristics and benefits ..... 3
- Figure 5: Classical software application lifecycle ..... 4
- Figure 6: Enterprise phases and challenges with respective services and products..... 5
- Figure 7: Design Science Research Methodology ..... 6
- Figure 8: Research Structure..... 12
- Figure 9: Services overview..... 14
- Figure 10: Typical traditional application for an online bookstore ..... 16
- Figure 11: Transformed bookstore using BaaS and FaaS ..... 16
- Figure 12: Typical versus Serverless cloud costs..... 17
- Figure 13: Serverless versus traditional cost increment..... 17
- Figure 14: Function schematic ..... 23
- Figure 15: A Faasified application using a Microservice Architecture..... 23
- Figure 16: Performance differences of CSPs..... 25
- Figure 17: Capgemini Cloud Strategy Framework..... 30
- Figure 18: Action Priority Matrix..... 41
- Figure 19: Questionnaire ..... 44
- Figure 20: Results ..... 45
- Figure 21: Plan of Action..... 46
- Figure 22: Demonstration Spider Graph..... 48
- Figure 23: Demonstration Position Matrix ..... 49
- Figure 24: Consultant instructions improvement ..... 55
- Figure 25: Interviewee instructions improvement..... 56
- Figure 26: ESA tool - Frontpage..... 77
- Figure 27: ESA tool - Instructions page..... 77
- Figure 28: ESA tool - Overview page ..... 78
- Figure 29: ESA tool - Questionnaire page..... 78
- Figure 30: ESA tool - Results page ..... 79
- Figure 31: ESA tool - Plan of Action page ..... 79
- Figure 32: ESA tool - Next steps page..... 80
- Figure 33: ESA tool - Epilogue page..... 80
- Figure 34: ESA tool - Settings page..... 81
- Figure 35: ESA tool - Industry standard settings page..... 81



# Abbreviations



BaaS	
Backend as a Service.....	13, 15, 16, 23, 24, 26, 27, 47, 83
CSP	
Cloud Service Provider.....	13, 14, 15, 18, 19, 20, 22, 24, 25, 26, 27, 33, 34, 38, 40, 47, 49, 57, 70, 71, 73, 74, 75, 76, 83, 85
CxO	
Chief "X" Officer.....	1
ESA	
Enterprise Serverless Assessment	7, 8, 9, 10, 11, 29, 31, 33, 39, 40, 41, 42, 43, 47, 53, 54, 55, 56, 57, 59, 61, 62, 63, 77, 78, 79, 80, 81
FaaS	
Function as a Service.....	13, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 34, 47, 83
IaaS	
Infrastructure as a Service.....	14, 15, 19, 21, 24, 33, 38, 70, 71, 73, 75, 84
Invent	
Capgemini Invent.....	1, 2, 3, 4, 5, 6, 10, 29, 62
PaaS	
Platform as a Service.....	14, 15, 16, 19, 20, 21, 47, 71, 73, 75, 83, 84
SaaS	
Software as a Service.....	13, 14, 15





# 1. Introduction



## Identifying the problem and formulating the research plan

This thesis presents the research performed at Capgemini Invent that aims to enable enterprises to adopt Serverless. This chapter first introduces Capgemini Invent in Section 1.1. Section 1.2 identifies the opportunity we aim to capitalize on, after which we identify a research gap within this opportunity in Section 1.3. Section 1.4 discusses the research approach to fill this gap, and Section 1.5 presents the research design. The chapter ends with an overall research structure in Section 1.6.

## 1.1 Context

Capgemini Invent, the digital innovation, consulting, and transformation brand of the Capgemini Group, commissions this research (Capgemini, 2021). Their goal is to help CxOs envision and build what is next for their business by combining market-leading expertise in strategy, technology, data science and creative design. "Define what's next" is their motto, which underlines their continuous interest in innovations that led to the constitution of this research.

One of Capgemini Invent's teams is Business Technology. This team empowers modern enterprises, their people, organisation, and processes (Capgemini Invent, 2022). They accelerate the transition to agile operating models, enabling cloud transformation, technology innovation, and a digital workplace built on trust and security. They claim that scalable, agile, cloud-native solutions improve resilience and embed sustainability in operations. To do so, they support the transformation of organizations to align business and IT. Their Cloud group supports partnered enterprises in reaching their business goals by servicing their cloud-related needs. Their efforts to transition toward cloud-native solutions and mission to optimize is the starting point of this research.

## 1.2 Opportunity Identification

Capgemini Invent, hereafter Invent, supports enterprises in achieving business goals by serving their IT needs. Invent realizes ongoing partnerships with enterprises to support their innovation at a larger scale than temporary projects. Their consultants have already helped most customers invest in the traditional Cloud, which opens the opportunity to advance further. This opening, combined with Invent's mission to keep innovating, resulted in the cause for this research: to explore new cloud opportunities.

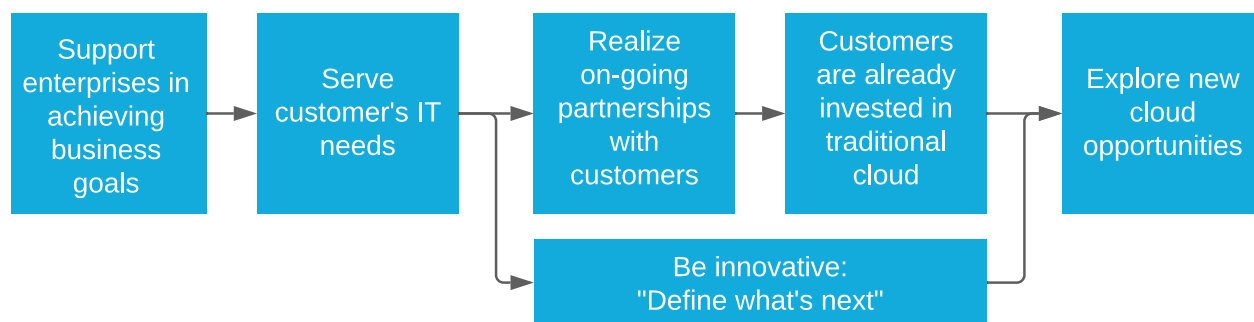


Figure 1: The research cause

Several requirements restrict the definition of a new cloud opportunity. Because Invent aims to support enterprises strategically, this cloud opportunity must support one or more of their typically pursued business goals. According to the literature, thirteen generalized goals illustrate typical enterprises (ISACA, 2018). Figure 2 shows these goals in an arbitrary order.

EG01: Portfolio of competitive products and services	EG02: Managed business risks	EG03: Compliance with external laws and regulations
EG04: Quality of financial information	EG05: Customer-oriented service culture	EG06: Business service continuity and availability
EG07: Quality of management information	EG08: Optimization of internal business process functionality	EG09: Optimization of business process costs
EG10: Staff skills, motivation and productivity	EG11: Compliance with internal policies	EG12: Managed digital transformation programs
EG13: Product and business innovation		

Figure 2: The thirteen Enterprise Goals (ISACA, 2018)

As Invent's promise towards their customer is to "define what's next", another requirement is that the cloud opportunity is a new addition to Invent's services and a relevant innovation as of September 2021. Invent aims for cloud-native solutions. These are solutions intentionally designed for the Cloud, but more specifically:

*"Cloud-native is a distributed, elastic and horizontally scalable system composed of (micro)services which isolate state in a minimum of stateful components. The application and each self-contained deployment unit of that application are designed according to cloud-focused design patterns and operated on a self-service elastic platform"* (Kratzke & Quint, 2017).

According to the literature, Serverless Computing is the capability that promises to advance the quality and development of cloud-native solutions (Gannon, Barga, & Sundaresan, 2017). Serverless promises to become the default computing paradigm of the Cloud era (Jonas, et al., 2019). When looking at recent publications, we state that the interest in Serverless has steadily grown and still is growing since its introduction in 2016, as shown in Figure 3 (Digital Science & Research Solutions, Inc., 2021). Surveys among enterprises conclude that 68% expect to adopt Serverless in the next two years (IBM Market Development & Insights, 2021). While Invent is not invested in Serverless's possibilities at this time of writing, and the literature emphasizes the relevancy of this innovation, the opportunity arises to investigate if Serverless is a valuable extension to Invent's services.

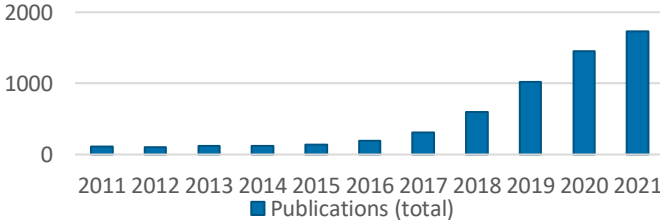


Figure 3: Serverless publications over the years (Digital Science & Research Solutions, Inc., 2021)

In short, Serverless architectures allow organizations to build applications without worrying about the applications' environment (Sewak & Singh, 2018). The organization outsources the management of these environments to another party. Outsourcing enables organizations to focus purely on the development of the code. Hence the word "serverless". While the code still runs on a server, it is no longer the concern of the organization to manage these servers; they enjoy a serverless experience. As further discussed in Section 2, Serverless allows less time spent managing servers, an invocation-based billing model, and automatic scaling. These characteristics result in a faster time to market, a lower total cost of ownership, more elastic and agile applications, improved security, and reduced energy consumption. Figure 4 shows that these characteristics and their respective benefits support the typical generalized enterprise goals. All but three of them can benefit from Serverless.

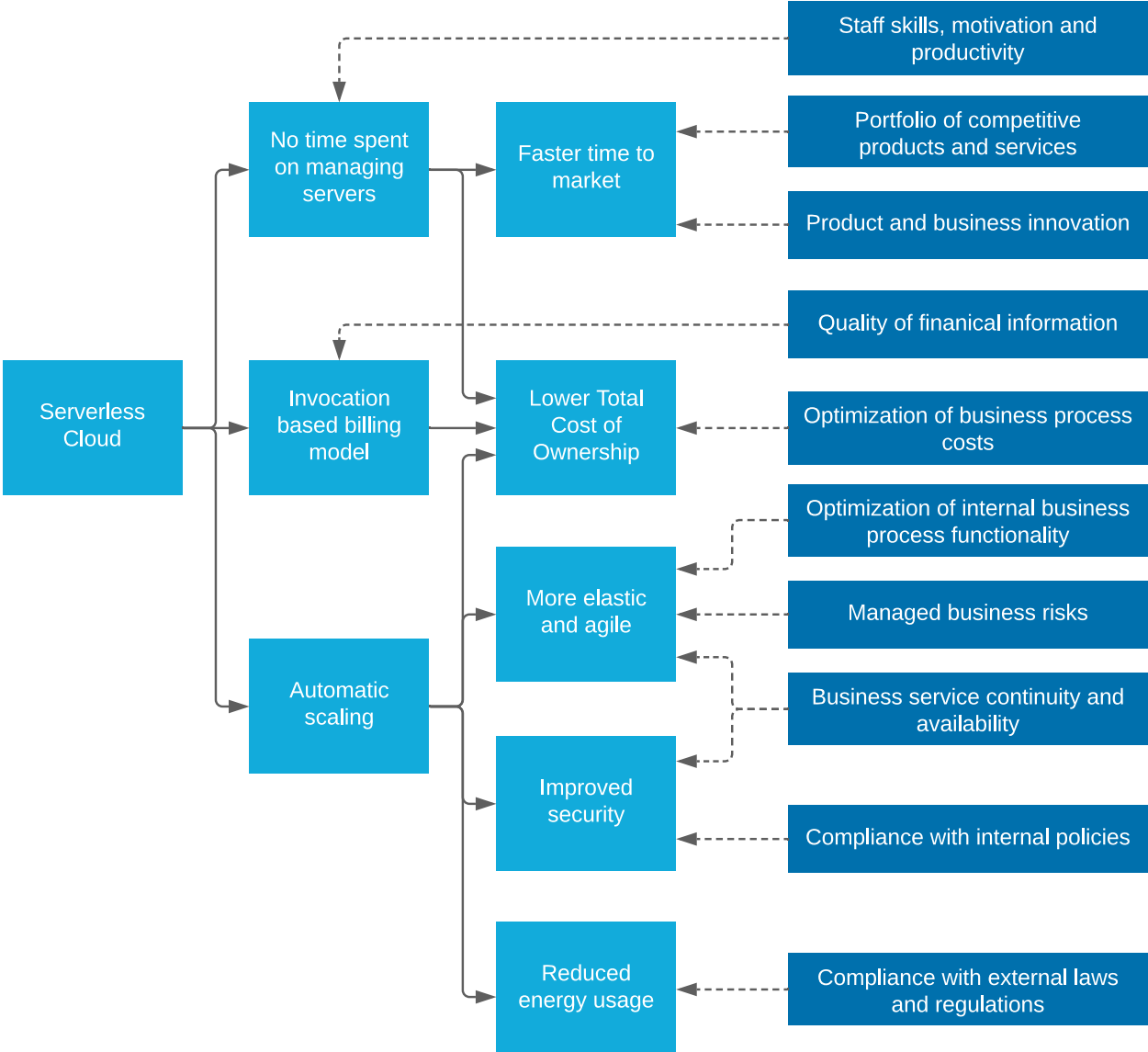


Figure 4: Mapping enterprise goals onto Serverless characteristics and benefits

To conclude, Serverless can strategically support enterprises and Invent has not explored its opportunities yet. Therefore it meets the requirements as the Cloud opportunity to further study in this research.

## 1.3 Research Gap Identification

We chose Serverless as the opportunity for this research because of the growing interest in academic literature and its alignment with the goals at Invent. For this opportunity, we need to determine what gap within the scientific literature to fill to further the scientific field of Information Systems concerning Serverless. By doing an exploratory literature review, we found that scientists have been working hard to develop frameworks to resolve issues around Serverless (Kritikos & Skrzypek, 2018). Preliminary research assessed these frameworks based on the classical software application lifecycle (Kritikos & Skrzypek, 2018). This lifecycle, shown in Figure 5, contains seven stages starting with the requirements analysis. This review focused on six phases and ignored the requirements analysis because they claim that the focus of existing cloud management frameworks does not include this initial phase (Kritikos & Skrzypek, 2018). Their claim about this scope is not further substantiated, leaving a research gap. We also did not find any relevant studies about requirements analysis concerning Serverless when doing further exploratory research. We deem this gap problematic as other sources state that determining if Serverless is the right fit for an organization is crucial for its success (Eivy & Weinman, 2017).

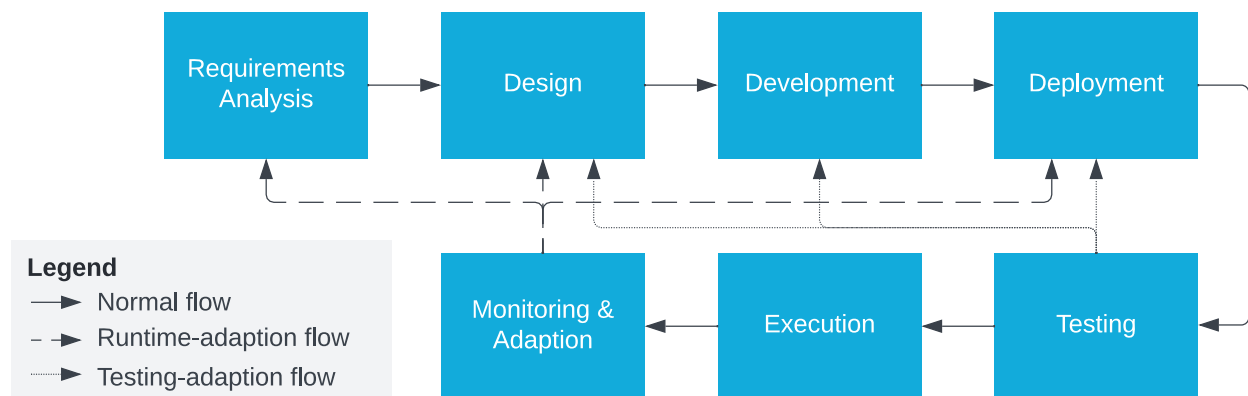


Figure 5: Classical software application lifecycle, based on (Kritikos & Skrzypek, 2018)

As stated in preliminary research, the impact of Serverless within scientific literature is limited (Stefanac & Colomo-Palacios, 2021). Previous studies countered this shortcoming by including grey literature within their research (Stefanac & Colomo-Palacios, 2021). They claim that scientists must include grey literature to answer research questions about Serverless (Stefanac & Colomo-Palacios, 2021). However, the downside of using grey literature is that it does not meet white literature's academic substantiation. This gap between the current state of Serverless within white and grey literature is the second research gap we discovered.

Within this research, we aim to fill both of these gaps. By scientifically substantiating the grey literature about Serverless, we aim to shape the currently missing requirements analysis phase. To ensure filling this gap is valuable, we align it with the needs of organizations. A market survey in 2021 claims that the main barrier to Serverless adoption within non-using enterprises is a shortage in guidance (IBM Market Development & Insights, 2021):

*"For current nonusers, primary concerns centre around a lack of clarity about how to move forward—finding relevant use cases, where to get executive support, attracting talent and needing tactical insights into the process."*

Because this guidance problem is present within all parts of the adoption process, but time constraints limit us, we scope the study to the adoption phase, where most of the enterprises currently are and distinguish four maturity phases:

1. The organization is not committed to Serverless.
2. The organization is committed to Serverless but has not started with the adoption.
3. The organization is committed to Serverless and actively adopting it.
4. The organization has successfully performed its Serverless adoption.

In each phase, with the exemption of the fourth, different challenges can arise for the organization. Based on unstructured interviews with experts at Invent, we expect that organizations might be:

1. Unaware if Serverless is a valuable investment for them or if they have the right capabilities.
2. Unfamiliar with developing in a Serverless way.
3. Stuck in advancing their Serverless adoption.

Solving these challenges benefits organizations and is therefore deemed an opportunity for Invent to serve its customers. Invent can start assessing their customers' strategy to verify whether Serverless is a suitable innovation for them, provide development guidance, and perform maturity assessments to determine how an enterprise can progress its adoption. Each of these requires a new artefact. The first opportunity requires a Serverless strategy assessment, the second a Serverless development roadmap, and the third a Serverless maturity framework. Figure 6 summarizes these opportunities.

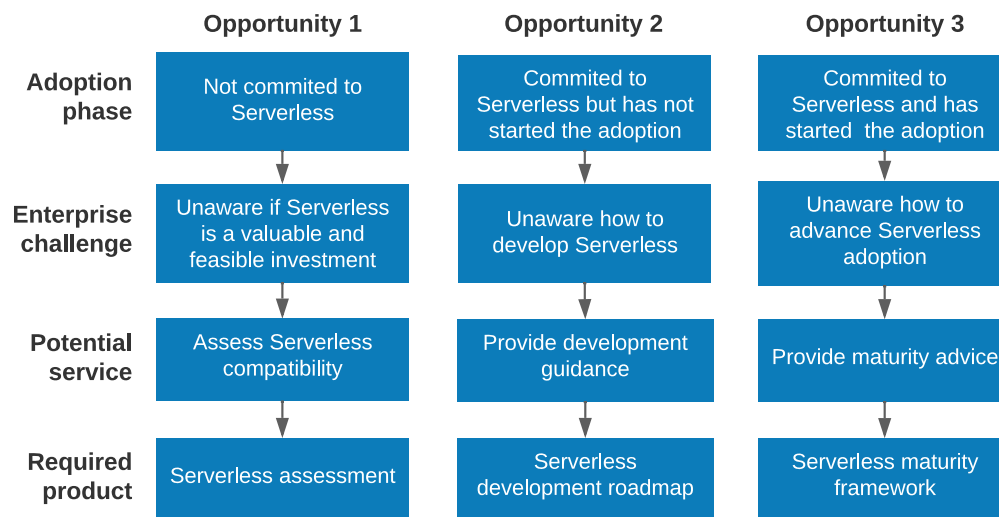


Figure 6: Enterprise phases and challenges with respective services and products

To determine which opportunity to tackle in this research, Heerkens (2017) suggests choosing the one with the most significant impact at the lowest cost (Heerkens, 2017). Within our study, impact contains two factors:

1. It (partly) fills the research gap concerning the requirements analysis phase within existing Serverless frameworks.
2. It enables Invent to provide new or improved services to its customers.

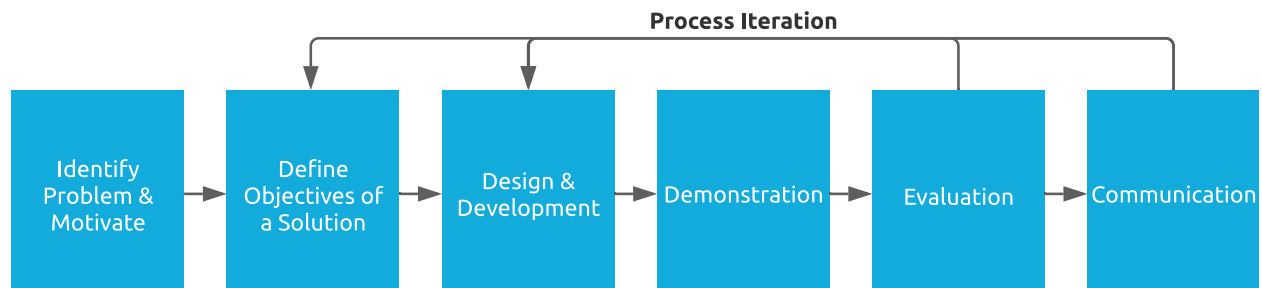
When looking at the goal of the requirements analysis phase, we find: “The purpose of the Requirements Analysis Phase is to transform the needs and high-level requirements specified in earlier phases into unambiguous (measurable and testable), traceable, complete, consistent, and stakeholder-approved requirements.” (Maryland.gov, 2022). Looking at this definition, we believe it is essential to determine what high-level requirements organizations can meet using Serverless. In other words, we expect the requirements analysis phase at the scale of Invent’s customers to be mainly about finding the fit between Serverless technology and desired business value. Therefore, we deem an initial Serverless suitability assessment the best fit. On top of that, interviews with Invent’s consultants show that Invent’s customers are primarily in the first phase of their Serverless adoption, making the requirements analysis phase relevant for Invent. Therefore, the first opportunity is the most promising for both factors and is thus further investigated within this research. Because Invent provides management consulting for enterprise-scale organizations, we scope the research goal towards enterprises. This decision leads us to the following research question for this thesis:

*How can an enterprise assess its fit with Serverless technology?*

We present the approach to answering this question and the corresponding results throughout the rest of this document.

## 1.4 Research Approach

To execute this research appropriately, we follow a proven research methodology. Since we aim to develop a Serverless assessment, this methodology must be a design science research paradigm that focuses on realising artefacts instead of empirical research (Hevner, March, Park, & Ram, 2004). Therefore, we use the DSRM, the Design Science Research Methodology (Peppers, Tuunanen, Rothenberger, & Samir, 2007). We choose the DSRM because it focuses on information systems. Figure 7 shows the six phases of the DSRM, starting with problem identification and motivation.



*Figure 7: Design Science Research Methodology*

Next, we discuss each of the phases. We declare sub-questions that need to be answered during that phase to ensure we gather the required knowledge.

### Phase 1: Identify Problem & Motivate

The first phase defines the research problem and justifies the value of a solution (Peppers, Tuunanen, Rothenberger, & Samir, 2007). Within this research, we have already identified the problem in Section 1.2. This identification led to the research question that we aim to answer with this research approach. We, therefore, consider this phase to be completed and define no further sub-questions.

## Phase 2: Define Objectives of a Solution

The second phase infers the objectives of the solution (Peffer, Tuunanen, Rothenberger, & Samir, 2007). Peffer (2007) claims that this requires: "knowledge of the state of problems and current solutions and their efficacy". Within this research, we must determine the current state of Serverless within the literature. We first formalize the definition of Serverless to scope the study appropriately. Hereafter we need to find motives for why an organization would want to use Serverless to determine if enterprises benefit from its adoption. Then we need to define when Serverless is the right choice for an organization to decide if it fits the organization. Lastly, we determine how the technology works to determine if the organization has the right capabilities to use it. To gather this theoretical framework, which we refer to as the current state of the art of Serverless, we define the following sub-question:

*Sub question 1: What is the current state of Serverless in literature?*

- a. What is the definition of Serverless?*
- b. Why is Serverless a valuable investment?*
- c. When is Serverless the right technology to use?*
- d. How does Serverless technology work?*

We present the answer to this question in Chapter 2 of this document.

## Phase 3: Design & Development

The third phase designs and creates the artifactual solution (Peffer, Tuunanen, Rothenberger, & Samir, 2007). We anticipate this artifactual solution to be an enterprise assessment for using Serverless. To realize this, we first develop a theoretical version based on the literature results of phase 2. Hereafter, we complement the academic knowledge by adding practical knowledge to get a well-founded assessment. We call this assessment the Enterprise Serverless Assessment, in short, ESA. To realize the ESA, we need to answer the following sub-question:

*Sub question 2: How can we create an Enterprise Serverless Assessment?*

- a. What areas within an enterprise do we need to assess for a comprehensive Cloud strategy?*
- b. What does the literature state as criteria for each of these areas concerning Serverless?*
- c. What do experts state to be the criteria for each of these areas concerning Serverless?*
- d. What criteria must the organization prioritize during adoption?*
- e. How can a consultant assess these criteria?*

We present the answer to this question in Chapter 3 of this document.

## Phase 4: Demonstration

The fourth phase demonstrates the efficacy of the artefact to solve the problem (Peffer, Tuunanen, Rothenberger, & Samir, 2007). The ESA is applied to an enterprise organization to validate its performance. This results in the following sub-question:

*Sub question 3: What results does the ESA deliver when assessing an enterprise?*

*Sub question 4: What actions does the ESA advise when assessing an enterprise?*

We present the answers to these questions in Chapter 4 of this document.

## Phase 5: Evaluation

The fifth phase observes and measures how well the artefact supports a solution to the problem (Peffer, Tuunanen, Rothenberger, & Samir, 2007). We can determine the ESA's effectiveness and discover potential improvements and pitfalls by evaluating the demonstration results. Therefore, we answer the following three sub-questions:

*Sub question 5: Is the ESA usable?*

- a. Is the ESA methodology appropriate for an enterprise organization?*
- b. Is the time required within reasonable limits for an enterprise organization?*
- c. Are the assessment topics comprehensive?*
- d. Are the assessment criteria relevant?*
- e. Is the ESA in line with general prior expectations?*
- f. Are the ESA's results in line with an organization's self-perception?*

*Sub question 6: Is the ESA effective in improving Serverless adoption?*

- a. Does the ESA provide insights regarding an organization's fit with Serverless?*
- b. Does the ESA provide advice that improves Serverless adoption?*

*Sub question 7: What limitations does the ESA have?*

We present the answers to these questions in Chapter 5 of this document.

## Phase 6: Communication

The final phase communicates the study to researchers and relevant audiences (Peffer, Tuunanen, Rothenberger, & Samir, 2007). We publicize the results through this thesis and a presentation. To finalize, we answer three closing sub-questions:

*Sub question 8: What conclusions can we draw?*

*Sub question 9: What recommendations can we give to the commissioning organization?*

*Sub question 10: What further research do we require?*

We present the answers to these questions in Chapter 6 of this document.

# 1.5 Research Design

To answer the research and sub-questions stated in Sections 1.2 and 1.4, we perform several different research methods throughout the study:

## Multivocal Literature review

Firstly, we perform a Multi Vocal Literature Review to create Serverless's current state of the art during phase 2 of the research. This method enables us to collect the available theoretical knowledge regarding Serverless, which is required to develop the initial version of the ESA. We use a multivocal version because preliminary literature reviews regarding Serverless found that alternatives were insufficient to answer their research questions (Sadaqat, Colomo-Palacios, & Knudsen, 2018) (Stefanac & Colomo-Palacios, 2021). This study needs a multivocal approach due to Serverless's novelty and limited impact in white literature (Stefanac & Colomo-Palacios, 2021). This method includes all relevant articles in both scientific



and grey literature. The grey literature is required because the available scientific papers lack knowledge within organizations. These organizations publish this information primarily through their website, whitepapers, and blogs that do not meet scientific standards but provide important practical insights.

### Delphi Study

Secondly, to extend the theoretical version of the ESA with practical knowledge, we perform a Delphi study in phase 3. A Delphi study enables us to improve the ESA by presenting it to a panel of experts that provide their opinion during several rounds (Barrett & Heale, 2020). After each round, we improve the artefact based on the findings. We then present it again to the experts to validate the changes. This validation ensures that a single expert cannot undermine the validity of the ESA because their input is peer-reviewed by the other experts. The goal is to let the experts come to a consensus regarding the contents of the ESA. We specifically choose this method because it enables us to fill the second research gap. After all, the experts can substantiate the grey literature. Because of this substantiation, we can close the gap between the grey and white literature. On top of that, a Delphi study has worked for similar studies in the past (Van Dijk, 2017).

### Expert Panel

We must gather a diverse panel of experienced experts for the Delphi study. To ensure that the ESA remains scientifically relevant, we need experts within and outside the commissioning organization. On top of that, we need experts from all organisational layers to ensure it is comprehensive: from strategic to operational. Within our context, this means managers, preferably Chief Technology Officers due to their focus on technology, architects due to their bridging role between the organizational strategy and the operations, and developers as they are the operational employees within the IT process of the organization.

We found five experts who met these requirements and said they were willing to participate in the panel after requesting them to join through email. These experts remain anonymous as prescribed by the Delphi guidelines. Their organizational roles are:

- Global CTO
- CTO for Microsoft offerings and Microsoft MVP
- Enterprise Architecture director and consultant
- Architect and app modernization expert
- Architect and developer

Of these people, two are directly from within the commissioning organization, two others are from affiliated organizations, and one is from an unaffiliated organization.

### Rounds

We first ask the experts openly to suggest what they deem relevant for each assessment topic during the first round. These open questions are essential to prevent tunnel vision. Hereafter we present the findings in the literature and ask them if they agree or if changes are needed. When we have interviewed all experts, we combine all their input and create a new version. Parts can be added, removed, modified, or remain unchanged in this new version.

During the second round, we validate the changes of the first round and see if further changes are required. Once again, we interview each expert and present the new version of each assessment topic.

We ask if the expert agrees with each alteration or if further changes are necessary. On top of that, we ask what parts they believe an organization must prioritize, as further discussed in Section 3.4. After we interviewed all experts for the second time, we update the ESA once more.

After the second round, we verify if the experts have reached a consensus. We have a consensus when only minor changes occur during the second round. These changes may not include new additions or alterations that change the original intention. When this is not the case, we organize the third round. We deem three rounds the maximum number possible within this research's time restrictions.

#### Interview Method

We perform each round of the Delphi study through interviews. These sessions take 1 hour each and take place through online Microsoft Teams calls. We interview the experts separately and do not inform them about the other experts to prevent competitive behaviour. Each round starts with an explanation about the goal of the round and what questions we will ask. We prepare and present a PowerPoint deck to support the session. This deck contains the explanation and the discussed subjects. If the expert agrees, we record each session. After the meeting, we review the recording and process the input of each expert individually. After each interview is processed, they are combined to create a new ESA version.

#### Case Study & Expert Opinion

Lastly, we perform a case study during phase 4 to determine the efficacy of the artefact and to what extent it helps solve the problem as prescribed by Wieringa (2014). During this case study, we apply the ESA to one of Invent's customers within practice conditions. Because it is impossible to perform the ESA and measure the long-term outcomes within the time constraints of this research, we choose the organization based on its Serverless maturity. We look for an organization that has already invested a lot in its Serverless adoption and has accumulated a lot of insights into the process. This way, we can ask a representative if they can identify with our findings and judge if the advised actions have worked for them in the past.

#### Organization

We reached out to three organizations, of which one agreed to participate, another organization responded but could only fill in surveys, and the last organization did not respond. Because surveys do not allow for follow-up questions, we continue only with the first organization, which is satisfactory, according to Wieringa (2014). This organization is a multinational financial service provider with an estimated number of about 50.000 employees and has been working with Serverless for multiple years. They have experienced architects and developers to validate if our findings align with their experience and self-perception.

#### Method

The case study takes place during two sessions over Microsoft Teams calls, and we record it when the organization gives permission. We fill in the questionnaire with an organisation representative during the first round. The representative is an architect within the organization and states to be acquainted with the required information. We present the results during the second round and advise on steps that would advance their Serverless adoption. Hereafter we ask them to respond to our findings to evaluate the ESA.

## 1.6 Research Structure

We structure this thesis according to the phases of the DSRM, as advised by Peffers et al. (2007). By following the DSRM structure, we use a proven communication method that provides clarity to the reader (Peffers, Tuunanen, Rothenberger, & Samir, 2007). We combine the DSRM, the research goals, the discussed research methods, and the proposed deliverables to create a comprehensive research plan. We visualized this plan in Figure 8, available on the next page. To sum up, we discuss the following matters within the chapters:

Within the first chapter, we identify the research opportunity to capitalize on and a research gap to fill during this study. In the second chapter, we gather available knowledge about Serverless in grey and white literature to get a well-founded base for the rest of this research. We need this knowledge to create the first version of the ESA that we base purely on theory. In the third chapter, we let experts criticize the findings in the literature and complement the theoretical knowledge with the practical knowledge of experts. We translate the results into criteria within the ESA and structure them within a tool. In the fourth chapter, we demonstrate the usability of the ESA within practice conditions by assessing an organization. This demonstration provides us with results that we evaluate in the fifth chapter to determine the effectiveness of the ESA in delivering the guidance that enterprises need to adopt Serverless. The last chapter contains the conclusion, recommendations for the organization, and proposals for future research.

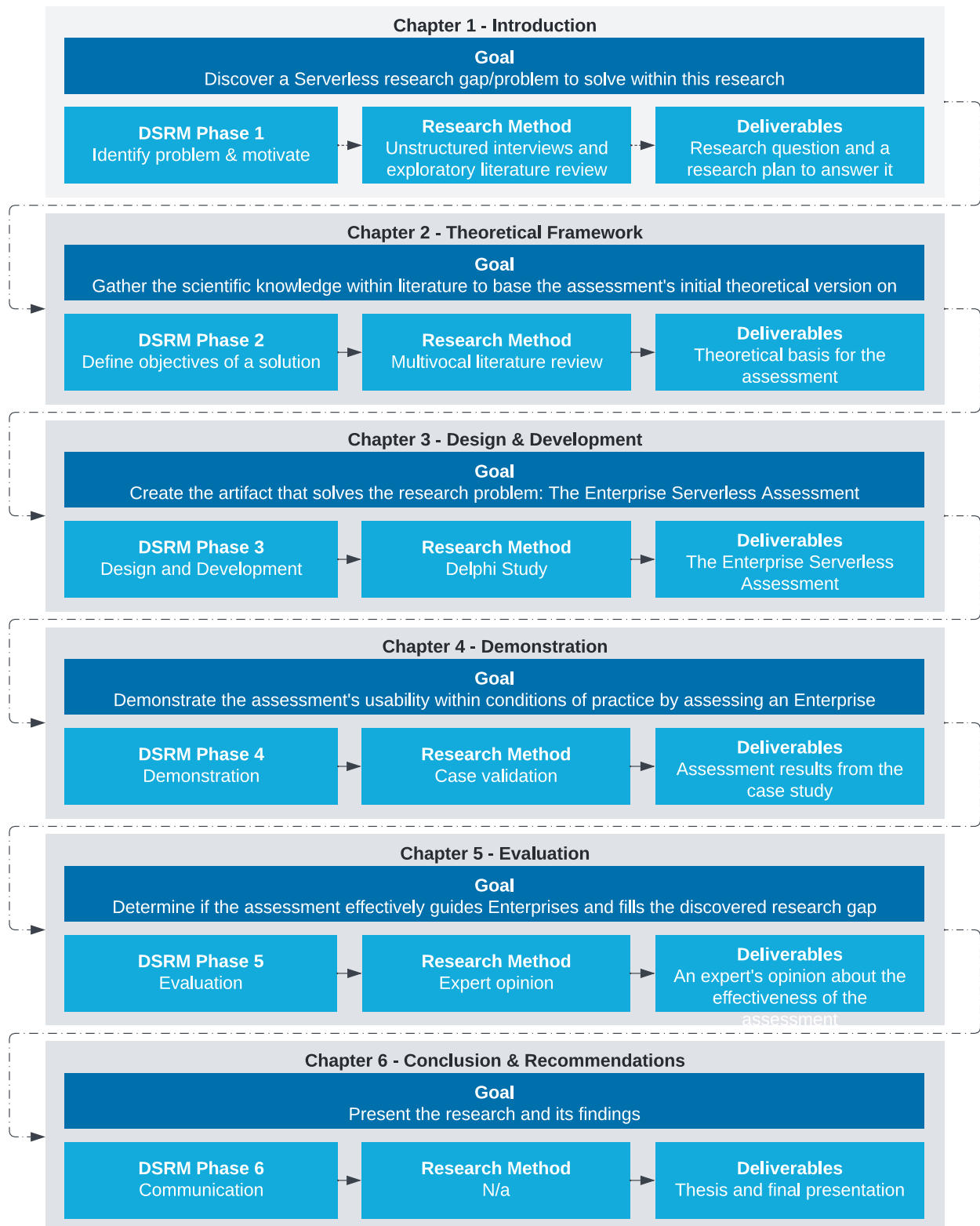


Figure 8: Research Structure

## 2. Theoretical Framework



### Gathering the theoretical knowledge needed to start the development

This chapter presents the state of the art of Serverless, which results from the performed literature study. The state of art functions as the theoretical framework for this research and leads to the artefact's first version. The chapter starts by defining the term Serverless in Section 2.1. Section 2.2 discusses the promise of Serverless and why enterprises want to use it. Section 2.3 describes what factors influence the fit of Serverless for an organization or application. Section 2.4 explains how the technology works. The chapter ends with a summary in Section 2.5.

### 2.1 Definition

Serverless promises that it allows organizations to build applications without worrying about their applications' environment (Sewak & Singh, 2018). The organization outsources the management of these environments to another party. Outsourcing enables organizations to focus purely on the development of the code. Hence the word "serverless". While the code still runs on a server, it is no longer the concern of the organization to manage these servers; they enjoy a serverless experience.

Researchers seem to have different definitions for the term Serverless. During a survey among experts, 58% claimed that "serverless" describes the so-called "Function as a Service" offering (Leitner, Wittern, Spillner, & Hummer, 2019). Function as a Service, hereafter FaaS, is both a cloud service model (Soldani, Yussupov, Breitenbücher, Brogi, & Leymann, 2020) and a computing architecture (Sewak & Singh, 2018). The first one refers to a service offered by cloud providers such as Amazon's AWS or Microsoft's Azure (Soldani, Yussupov, Breitenbücher, Brogi, & Leymann, 2020). The second one defines an architecture for IT applications that uses such services (Sewak & Singh, 2018). Hence, the cloud service model enables the architectural style.

The Serverless definition survey showed that 35% of the experts claim it describes all cloud offerings that do not require managing servers (Leitner, Wittern, Spillner, & Hummer, 2019). This definition implies that other offerings meet the serverless characteristics. Literature most often describes serverless as the combination of Function as a Service and Backend as a Service, in short, BaaS (Nupponen & Taibi, 2020). BaaS is even more serverless than FaaS in that the organization no longer needs to deliver code for the functions but can use off-the-shelf services serviced by a provider (Nupponen & Taibi, 2020). BaaS requires less effort but is limited to functionality serviced by the CSP and might therefore not suit tailored needs. FaaS and BaaS together create an exciting combination where the organization reuses whenever possible through BaaS and uses FaaS when more tailored functionality is required. An example of a BaaS service is Google Firebase, a wholly managed database that can directly integrate into applications; Firebase manages all data on behalf of the organization (Nupponen & Taibi, 2020). One other cloud service delivers a Serverless experience: Software as a Service, in short, SaaS. SaaS provides the customer with a complete application; they do not need to install or maintain anything (Ahmad, Naveed, & Hoda, 2018). Because SaaS existed long before the rise of Serverless, it is often not included when discussing Serverless in the scientific literature; it is an individual subject. Therefore, this study does not include SaaS in the definition of Serverless and focuses on FaaS and BaaS.

As mentioned, an external party provides the serverless experience for an organization. This external party is the Cloud Service Provider; in short, CSP (Rajan, 2020). Organizations hire them to manage their servers. Here originates the phrase "as a service"; the CSP provides the required services to other organizations. The most used services are Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS) (Kächele, Spann, Hauck, & Domaschka, 2013). These services differ in the layers of IT infrastructure that they supply. As shown in Figure 9, each of the different services moves more responsibility from the organization to the CSP. The figure starts with "on-premise", the traditional format where an organization manages all parts of the infrastructure itself. The servers are literally on their premise, hence the name. Because buying and maintaining servers is expensive, complex, and risky, organizations started outsourcing these tasks. They purchased them as a service from other organizations. Because the servers were no longer on the premise, they were said to be in the Cloud, somewhere in the air where it is no longer their concern.

On Premise	IaaS	PaaS	Serverless		SaaS		
			FaaS	BaaS			
Application	Application	Application	Client	Code Execution	Client	Code Execution	Application
Runtime	Runtime	Runtime	Runtime	Runtime	Runtime	Runtime	Runtime
OS	OS	OS	OS	OS	OS	OS	OS
Virtualization	Virtualization	Virtualization	Virtualization	Virtualization	Virtualization	Virtualization	Virtualization
Networking	Networking	Networking	Networking	Networking	Networking	Networking	Networking
Storage	Storage	Storage	Storage	Storage	Storage	Storage	Storage
Hardware	Hardware	Hardware	Hardware	Hardware	Hardware	Hardware	Hardware

Legend      Customer Managed      Provider Managed

Figure 9: Services overview, based on (Wolf, 2021)

For moving applications to the Cloud, organizations can perform one of six actions on their applications, the 6 R's (Ahmad, Naveed, & Hoda, 2018):

1. *Rehost*

When rehosting an application, it moves from a local on-premises server to a new host located at a CSP. Nothing changes except the server's physical hardware that runs the application, also known as the infrastructure, and the server's geographical location. Lift & shift is a commonly used term for this transition (Logicata, 2021). Because the CSP now hosts the application, the organisation does not need to manage the infrastructure. They purchase their Infrastructure as a Service (IaaS). When using IaaS, the service provider governs all physical parts in one of their data centres. This service includes the virtualisation through which multiple customers of a CSP can use the same physical hardware. The CSP takes care of the location of the servers as well as updates and the required maintenance.

## 2. *Replatform*

During a replatform, the application also moves from an on-premises location toward the servers of a CSP but skips the operating system and runtime (Ahmad, Naveed, & Hoda, 2018). This migration creates the opportunity to move to a more recent version or a more suitable alternative (Logicata, 2021). When doing a replatform, the organization stops managing the platform and purchases the Platform as a Service (PaaS). PaaS provides the customer with an OS and runtime for their applications on top of the IaaS offerings. The CSP installs them and keeps them up to date.

## 3. *Repurchase*

An organization can choose to repurchase whenever a solution is available in the Cloud that delivers similar functionality as theirs (Ahmad, Naveed, & Hoda, 2018). The organization then purchases the Software as a Service (SaaS). Examples of SaaS are applications like Gmail and YouTube.

## 4. *Retire*

Organizations can choose to retire an application whenever it is no longer needed (Ahmad, Naveed, & Hoda, 2018). When moving an application to the cloud or leaving it on-premises results in no additional value, the organization shuts it down.

## 5. *Retain*

If moving an application to the cloud delivers no additional value or results in high costs, an organization can choose to keep it on-premises and retain it (Ahmad, Naveed, & Hoda, 2018). The situation remains unchanged.

## 6. *Refactor / Re-architect*

Whenever an application is unsuitable to be moved to the cloud as-is, is experiencing problems in maintenance or development, or no longer meets the required standards, the organization can choose to refactor (Ahmad, Naveed, & Hoda, 2018). During a refactor, the organization modernizes the application and builds it into a cloud-native application. Architects design these applications specifically to operate in the Cloud. During a refactor, Serverless becomes an option.

When using Serverless, specifically FaaS, architects break down the organization logic into functions (Sewak & Singh, 2018). Each of these functions works through a small, reusable code that executes whenever the respective operation is triggered. This trigger or event-based computing enables lower-cost computing that is more efficient since resources only need to be allocated to the function when initiated, ensuring these resources are not sitting idle whenever the function is not required. The CSP manages the allocation and scaling of these resources based on the event triggers. All major cloud service providers have FaaS service models available. These providers are Amazon's AWS Lambda, Microsoft's Azure Functions, Google Cloud's Functions, and IBM's Cloud Functions (Sewak & Singh, 2018). So, while the organization still maintains the front-end client that triggers the functions or services and delivers the code, execution is no longer their concern.

Organizations can transform traditional applications into FaaS/BaaS enabled ones in the following way. As shown in Figure 3, a conventional application contains a client running in a user's internet browser, which connects to an external server. This server has all functionality to handle the requests within a single

system, a so-called monolith, and stores all data in a database. The organization manages and maintains all parts of the application.

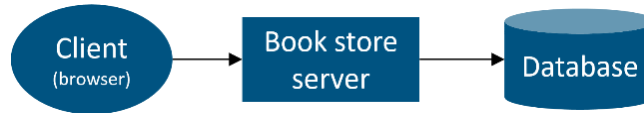


Figure 10: Typical traditional application for an online bookstore, based on (Roberts, 2018)

An architect can transform the traditional application into the one shown in Figure 11. This FaaS/BaaS enabled application keeps a client within the user's internet browser but stops sending each request to a single server. The server and database are split up and no longer managed by the organization:

- The BaaS service Amazon Cognito now handles user authentication (Amazon Web Services, 2021).
- The BaaS service Google Firebase now stores and manages the data (Google, 2021).
- The bookstore's tailored functionality now consists of a purchase function and a search function hosted by the FaaS service AWS Lambda (Amazon Web Services, 2021).
- These functions trigger through Amazon's API gateway, another BaaS service (Amazon Web Services, 2021).

In this scenario, the organization only needs to realise and manage the client and the code for the purchase and search functions. The CSPs handle all others.

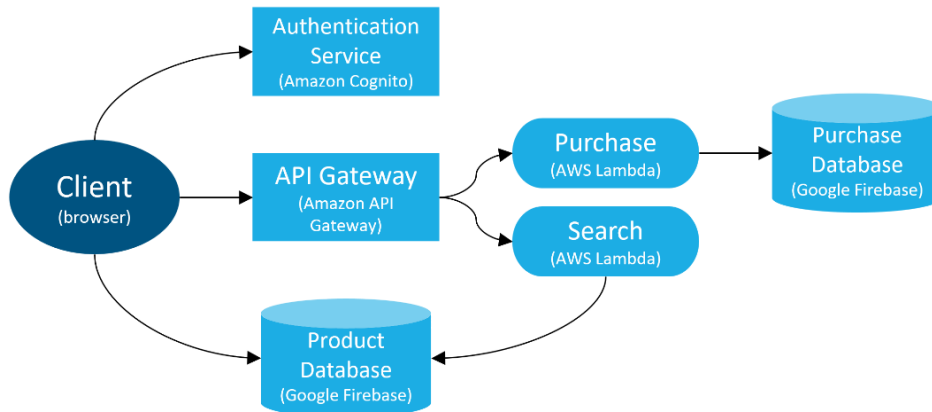


Figure 11: Transformed bookstore using BaaS and FaaS, based on (Roberts, 2018)

## 2.2 Promise

The promise of Serverless, according to literature, is to provide organizations with a disruptive edge, optimal efficiency, advanced agility and save costs (Sewak & Singh, 2018). On top of that, it allows for new architectural styles and improved security. This section discusses these benefits and what causes them.

### 2.2.1 Saving costs

In a case study mentioned by Michael O'Connor, the Chief Architect at The Coca-Cola Company, a move from PaaS to FaaS reduced the cost of handling around 360 million yearly requests from \$12,864 to \$4,490 (Goncharov, 2017) (Amazon Web Services, 2020). Organizations can achieve such cost reductions through



FaaS due to its pay as you go model: The organization pays per single use of the function. Pay as you go eliminates the costs of hosting the function when not using it while still delivering similar performance to alternatives (Albuquerque Jr., Silva Ferraz, Oliveira, & Galdino, 2017). Pricing becomes flexible, and infrastructure costs reduce because you only purchase what you need (Sewak & Singh, 2018). Other cloud services require organizations to pay for the resources upfront based on demand forecasts (Tayal, et al., 2019). These forecasts carry an uncertainty that creates the need to purchase extra capacity as buffers. In FaaS, there is no need to invest in additional resources because they are scaled by the cloud service provider based directly on the actual demand, as visualized in Figure 12. The difference between the demand and the bought capacity is a cost opportunity that FaaS aims to capitalize on.

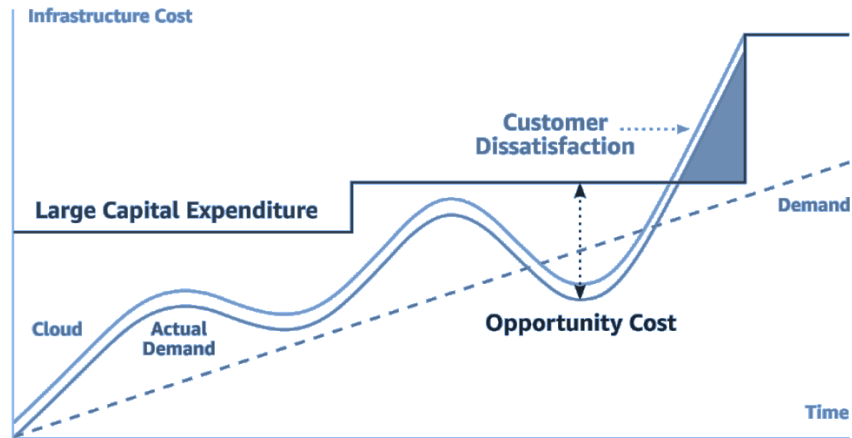


Figure 12: Typical versus Serverless cloud costs (Tayal, et al., 2019)

Because the prices in FaaS scale with the number of requests instead of the number of allocated instances, the increment is more linear (Albuquerque Jr., Silva Ferraz, Oliveira, & Galdino, 2017). Figure 13 shows how this linearity results in a lower opportunity cost for Serverless.

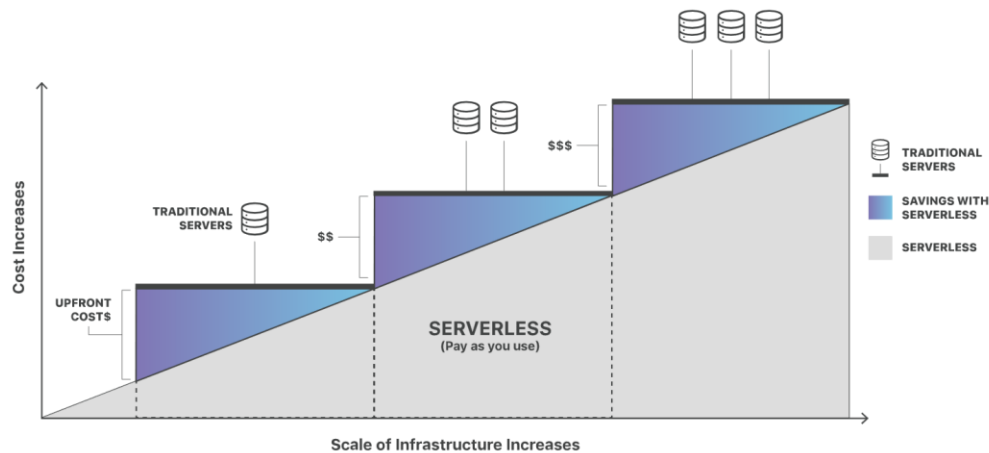


Figure 13: Serverless versus traditional cost increment (Cloudflare, 2022)

There are cases where a shift to FaaS did not lead directly to a decrease in infrastructural costs (Hellerstein, et al., 2019). Not every application has the same traffic and functionality, which causes a significant variance in results (Cui, 2019). Fortunately, this variance does not mean that those applications cannot save costs by moving to FaaS. When an organization wants to determine the actual cost savings of

FaaS, it must include all relevant fees, not just the infrastructure costs (Lefèvre, 2020). The so-called Total Cost of Ownership (TCO) (Lefèvre, 2020):

$$\text{Total Cost of Ownership} = \text{Infrastructure Costs} + \text{Development Costs} + \text{Maintenance Costs}$$

The TCO consists of three types of costs:

1. Infrastructure costs: The fees incurred by a Cloud Service Provider for hosting the application workload, also known as the cost to run (Tayal, et al., 2019).
2. Development costs: The initial costs for developing the application on a cloud service, also known as the cost to achieve (Tayal, et al., 2019).
3. Maintenance costs: The daily operation costs for running and maintaining the application, also known as the cost to support (Tayal, et al., 2019).

Literature quantifies the development costs based on the time and effort required for pre-planning the build of the application (Tayal, et al., 2019). Developers must determine future IT challenges upfront to prevent waste because of over-provisioned resources or customer dissatisfaction due to inadequate resources. The developers' time spent forecasting demand, provisioning auto-scaling, setting up network and load balancers, planning for availability, and purchasing licenses and software becomes unnecessary (Tayal, et al., 2019). This decline in workload reduces the required personnel and respective salary costs. Because serverless applications leverage event-based architectures, development teams can start directly with the development instead of planning a deployment architecture upfront (Tayal, et al., 2019). Research shows that serverless applications take only eight days to deploy, whereas a traditional cloud transformation takes about 25 days to deploy, a drop of 68% (Tayal, et al., 2019).

The maintenance costs consider the time and resources spent on continuing tasks after the deployment in production is completed (Tayal, et al., 2019). We categorise these into four areas: provisioning and scaling, security implementation, patching and OS updates, and ongoing application operations (Tayal, et al., 2019). These ongoing application operations can be delivering new features, testing, verifying, monitoring and logging (Tayal, et al., 2019). These take about 66 hours per month (Tayal, et al., 2019). With serverless, the organization imposes these maintenance tasks onto the CSP. This shift of responsibility lowers the time spent by the developer to somewhere between 11 and 35 hours per month, depending on the application (Tayal, et al., 2019). A drop of 45% to 80% as the developer's focus shifts toward developing the core capabilities instead of reboots and reconfigurations of the servers (Tayal, et al., 2019). These quantifications result from two case studies, of which Table 1 presents all individual costs (Tayal, et al., 2019).

	Case 1		Case 2		Mean Difference	
	EC2	Serverless	EC2	Serverless	\$	%
<b>Costs /month</b>						
<b>Infrastructure</b>	\$790	\$1090	\$296	\$378	\$191	35%
<b>Development</b>	\$640	\$205	\$640	\$205	- \$435	- 68%
<b>Maintenance</b>	\$4096	\$2240	\$4096	\$2240	- \$1856	- 45%
<b>Total</b>	<b>\$5526</b>	<b>\$3535</b>	<b>\$5032</b>	<b>\$2823</b>	<b>- \$2100</b>	<b>- 40%</b>

Table 1: Cost comparison between EC2 and Serverless (Tayal, et al., 2019)

Additional research reports show 50-70% cost reductions when using FaaS (Villamizar, Garcés, Ochoa, & Castro, 2017), with some migrations from IaaS reducing as much as 95% (Adzic & Chatley, 2017). A survey among experts within the field of Cloud confirms this reduction; 71% answered that the total costs of FaaS are lower than its alternatives (Leitner, Wittern, Spillner, & Hummer, 2019). To conclude, Serverless shows to be an effective way to save costs. While the infrastructure costs were higher in some Serverless solutions, the total costs were lower, showing the importance of the total cost of ownership when considering Serverless.

## 2.2.2 Optimal efficiency

The second promise of FaaS is optimal efficiency which originates in its event-driven nature (Sewak & Singh, 2018). Whenever a function is triggered, the CSP starts the function to perform the required action until it is done and shuts down again. The functions remain shut down until they are activated. Keeping the function off ensures that they do not have idle time making them more efficient than using a PaaS solution that constantly runs (Albuquerque Jr., Silva Ferraz, Oliveira, & Galdino, 2017). The CSP can now use those resources for other workloads ensuring higher utilisation and thus less waste. Better utilisation of resources results in ecologically greener computing, lowering energy usage and supporting organizations to reach sustainability goals (Blamire, 2019).

FaaS solutions are also scaled per function, whereas alternatives only scale per service or application. This scaling enables the isolation of high-volume transactions, ensuring that only those parts of the application that require more resources get more capacity allocated, making it perform more efficiently.

## 2.2.3 Advanced agility

The serverless architecture enables the customer to shift the maintenance, provisioning, and scaling to the CSP, resulting in a better time-to-market and lower risks (Sewak & Singh, 2018). Because the CSP is responsible for scaling and maintaining the function's environment, the organization achieves an advanced form of agility. Whenever the functions are unused, the organization does not pay, and when the demand is unexpectedly high, the CSP scales the function appropriately, preventing unavailability. The organization is no longer prone to changes in demand lowering the respective risks.

Serverless also decreases time to market because the organization's developers can focus on developing the core business logic instead of setting up and maintaining the servers (Sewak & Singh, 2018). A low time to market benefits enterprises as they can deliver their services earlier and makes them more adaptive to change (Cui, 2019).

## 2.2.4 Security

Serverless's last frequently mentioned benefit is providing security benefits, such as resilience towards Denial of Service (DoS) attacks (Pekkala, 2019). The provider delivers (near) limitless scaling, which prevents the application unavailability targeted by such attacks. The only downside is that automatic scaling during an attack can result in high costs billed by the CSP (Pekkala, 2019).

Another benefit is that Serverless applications have a smaller attack surface. This surface is smaller because when a function completes an action, the CSP destroys the instance and potential infections (Wagner & Sood, 2016) (Pekkala, 2019). After the following function call, a new unaffected sample starts.

The final security benefit is that many security actions are shifted towards the CSP when using Serverless. This shift eliminates the need for an in-house team to perform tasks such as provisioning firewall licenses and host scanning (Tayal, et al., 2019). CSPs also have more financial resources to improve and maintain the security measures of the servers they provide for their customers. Surpassing those resources is virtually impossible for most organizations (Graw, 2021).

## 2.3 Suitability

This chapter describes what circumstances and applications are best suited for utilising Serverless. The chapter contains four sections discussing the application characteristics, workloads, security requirements, and vendor lock-in.

### 2.3.1 Application characteristics

Research has shown that FaaS enabled applications can perform like PaaS solutions in most scenarios (Albuquerque Jr., Silva Ferraz, Oliveira, & Galdino, 2017). This performance is satisfactory since the goal was not to increase performance but to lower risk and costs. There is, however, an essential factor that can reduce performance: The cold start.

When unused, the CSP shuts down the functions; this ensures that idle functions do not waste resources. Because the functions are off, they must boot up before executing; this is called a cold start. Compared to continually available functions, the time a cold start takes can be significant and can hugely influence the performance of an application (Albuquerque Jr., Silva Ferraz, Oliveira, & Galdino, 2017). This overhead averages on the user's side from 300ms to 24s, depending on the configuration (Manner, Endreß, Heckel, & Wirtz, 2018). Several factors cause this overhead:

- Programming language: Compiled languages such as Java are more prone to a cold start than interpreted languages such as JavaScript (Manner, Endreß, Heckel, & Wirtz, 2018). Java, for example, needs to be run in a Java Virtual Machine (JVM). Booting a Java function usually takes 2 to 3 times longer than a Javascript one.
- Deployment package size: In the case of applications using interpreted languages, a larger deployment package size results in a significantly higher cold start time (Manner, Endreß, Heckel, & Wirtz, 2018). For applications using compiled languages, this is not the case.
- CPU and memory: Assigning more resources limits the cold start for all applications. This reduction is due to the extra resources speeding up the setup of the execution environment. This result is most significant for compiled languages (Manner, Endreß, Heckel, & Wirtz, 2018).
- The last factor that is expected but not yet proven to influence the cold start is the number of dependencies (Manner, Endreß, Heckel, & Wirtz, 2018). A higher number of dependencies means more need to be loaded in before the execution.

These factors imply that FaaS is most suitable for functions written in an interpreted language, with a small deployment package size and few dependencies. Extra CPU and memory can lower the cold start whenever a compiled language is required. According to experts, FaaS supports both backed utilities and user-facing applications (Leitner, Wittern, Spillner, & Hummer, 2019). In the latter case, the discussed technical challenges are more critical due to the potential impact on the user experience (Leitner, Wittern, Spillner, & Hummer, 2019).

One of the promises of FaaS is a reduction in costs. Organizations can achieve this reduction because FaaS exposes them to a pay-per-use system instead of a monthly subscription fee. Research shows that, compared to a PaaS solution, FaaS can reduce costs, but it depends on several circumstances. In 2017, a comparison between the AWS PaaS solution Beanstalk and the AWS FaaS solution Lambda showed that the FaaS solution is cheaper in the case of short execution times (Albuquerque Jr., Silva Ferraz, Oliveira, & Galdino, 2017). Three scenarios, see Table 2, were examined with extended read operations and short write operations. When it was predominantly short write operations, the monthly cost dropped below the monthly subscription fee. In the case of an equal or higher number of extended read operations, the monthly costs surpassed the monthly subscription fees. This result shows that shorter execution times are most suitable when using a FaaS solution.

Ratio	50/50		70/30		90/10	
Quantity	21.600.000	21.600.000	30.240.000	12.960.000	38.880.000	4.320.000
Time	100ms	300ms	100ms	300ms	100ms	300ms
Operation	Write	Read	Write	Read	Write	Read
Lambda 265MB	\$13.32	\$31.33	\$18.65	\$18.80	\$23.98	\$6.27
Monthly Costs	\$44.65		\$37.45		\$30.25	
Beanstalk Subscription	\$33.86					
Difference	\$10.79		\$3.59		- \$3.61	

Table 2: FaaS and PaaS costs comparison (Albuquerque Jr., Silva Ferraz, Oliveira, & Galdino, 2017)

## 2.3.2 Workloads

Literature shows that FaaS is more suitable for unpredictable workloads (Albuquerque Jr., Silva Ferraz, Oliveira, & Galdino, 2017). Organizations cannot scale a PaaS or IaaS solution appropriately if the workloads are unpredictable; FaaS scales on-demand making unpredictable workloads no longer an issue.

Since the costs incurred for FaaS are dependent on the execution time of the functions, and the Cloud Service Provider limits the maximum execution time, the execution time of functions must be predictable (Albuquerque Jr., Silva Ferraz, Oliveira, & Galdino, 2017). Unpredictable execution times can result in high costs or poor performance, making them less suitable for FaaS solutions.

## 2.3.3 Security

Section 2.2.4 already discussed the security benefits of Serverless, but organizations must also consider some challenges. Firstly, each function or service is a potential point of compromise, which causes the need to secure all communications between the services and validate all inputs and outputs (Pekkala, 2019). These measures require skilled developers, especially when using confidential data.

The need for governance increases due to a possibly high number of functions (Pekkala, 2019). Organizations must monitor who owns what functions and what privileges each function needs (Pekkala,

2019). Governance can become a hassle when done incorrectly, so the organization must reserve the capacity to do it properly (Pekkala, 2019).

### 2.3.4 Vendor lock-in

The last matter to consider is the risk of vendor lock-in when using FaaS (Pekkala, 2019). While every type of Cloud service causes some vendor lock-in, designing an application conforming to the specific Serverless offerings of a CSP makes the application dependent and migrating to other CSPs gets complex (Pekkala, 2019). This complexity means that whenever an organization cannot accept such dependency, Serverless might not be the best fit. However, most enterprises already have a partnership with a CSP, making the vendor lock-in less of an issue.

## 2.4 Technology

This section describes how Serverless can be utilised and discusses the best practices. It starts by discussing the required architecture, hereafter the different cloud service providers and when to choose which and ends with some design choices for applications.

### 2.4.1 Architecture

Architectural changes to the application are required to utilise Serverless. In a traditional monolithic application, developers code all functionality into one package. When using Serverless, architects decompose the application into functions. This decomposition is called faasification. Each function is a separate piece of code that performs a single action. A client triggers the function when it is required. The extent to which an application is faasified depends on the proportion of decomposed business logic. In the case of a partly faasified application, functions deliver some of the business logic while the monolithic application still provides the rest. Organizations can migrate their monoliths to faasified applications either gradually or entirely at once (Leitner, Wittner, Spillner, & Hummer, 2019). A typical application or complementation consists of 5 to 15 functions (Leitner, Wittner, Spillner, & Hummer, 2019).

Each function must be a triggerable isolated action. When triggered, the function performs a rule-based action. This action can either be a single action or a chain that might invoke other parts of the service ecosystem. As shown in Figure 14, triggering happens through either package feeds, called reactive functions, or through API requests, called restful functions. When triggered, the CSP provisions resources and boots up the function. Hereafter, the CSPs event handler routes the event toward the function that executes the action. The function returns a response and shuts down again.

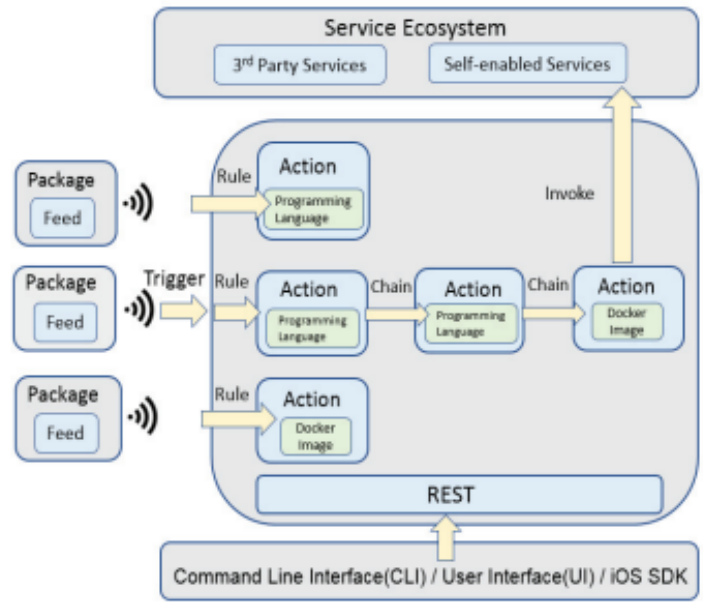


Figure 14: Function schematic (Sewak & Singh, 2018)

Experts state that building Serverless and FaaS applications requires a mental model that emphasises plugging together microservices (Leitner, Wittern, Spillner, & Hummer, 2019). Microservices are a proven way to realise a faasified application where the architect first decomposes the application into services (Sewak & Singh, 2018). Each service is independent and delivers part of the application's functionality. Separate teams can then develop the services without influencing others.

Serverless enables the decomposition of microservices into even more granular functions, often called nanoservices, which allows the organization to benefit from both architectures (Albuquerque Jr., Silva Ferraz, Oliveira, & Galdino, 2017). Figure 15 shows an example of such an application. Other methods that support adopting the Serverless mental model are functional programming and the immutable infrastructure paradigm (Leitner, Wittern, Spillner, & Hummer, 2019).

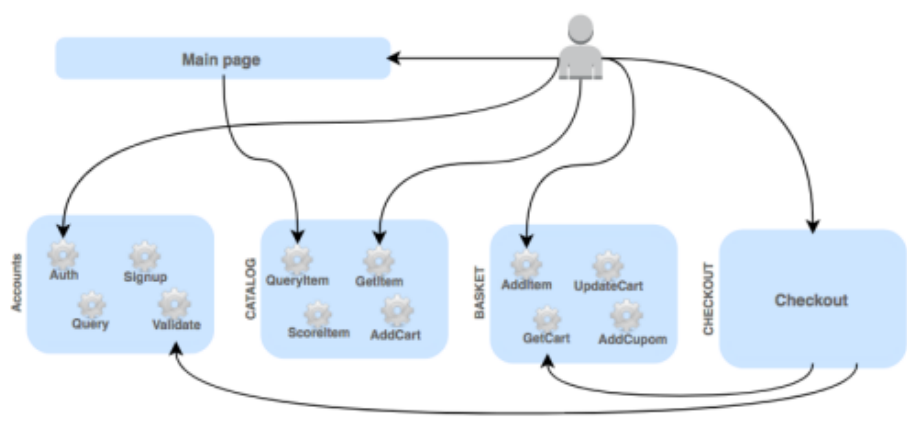


Figure 15: A Faasified application using a Microservice Architecture (Albuquerque Jr., Silva Ferraz, Oliveira, & Galdino, 2017)

A survey among experts shows that most Serverless applications utilise BaaS cloud services in conjunction with FaaS (Leitner, Wittern, Spillner, & Hummer, 2019). 78% of the experts use database services, 69%

use API Gateways, 66% use logging services, and 52% use IaaS services for the non-faaSified parts of applications (Leitner, Wittern, Spillner, & Hummer, 2019).

## 2.4.2 Cloud Service Providers

Choosing which CSP to host the FaaS functions is essential, as not all have the same functionality, performance, and pricing. Some decisive factors with respect to functionality and exemplary considerations are (Soldani, Yussupov, Breitenbücher, Brogi, & Leymann, 2020):

- Supported endpoints: Does the application require synchronous or asynchronous calls?
- Provided BaaS services: Are the required services available such as data storage, message queues, and logging?
- Supported function runtimes: Can they run our developers' preferred programming languages?
- Quota's: Do they allow the required package sizes and execution times?

The FaaS services of CSPs differ on many more factors than those four. One of the ways to choose the most suitable CSP is through FaaSter, a FaaS Platforms Selection Support System specifically designed to support researchers and practitioners (Soldani, Yussupov, Breitenbücher, Brogi, & Leymann, 2020). It is a web-based open sources application that enables multi-attribute queries to search for compatible CSPs (Soldani, Yussupov, Breitenbücher, Brogi, & Leymann, 2020).

The incurred costs are also heavily dependent on the chosen cloud service provider (Albuquerque Jr., Silva Ferraz, Oliveira, & Galdino, 2017). The case study previously discussed in section 2.3.1 tested the four most used CSPs, and as shown in Table 3, the results differed significantly (Albuquerque Jr., Silva Ferraz, Oliveira, & Galdino, 2017). IBM provided the cheapest option for this operation with a difference of 47% compared to the most expensive alternative (Albuquerque Jr., Silva Ferraz, Oliveira, & Galdino, 2017).

Provider	Read	Write	Monthly Cost
<b>AWS Lambda</b>	\$23.98	\$6.27	\$30.25
<b>Azure Functions</b>	\$23.33	\$6.05	\$29.38
<b>Google Functions</b>	\$33.53	\$7.72	\$41.25
<b>IBM OpenWhisk</b>	\$16.52	\$5.51	\$22.03

Table 3: Provider cost comparison (Albuquerque Jr., Silva Ferraz, Oliveira, & Galdino, 2017)

Billing in FaaS has two components: the number of function invocations and resource consumption (Bortolini & Obelheiro, 2020). The first is simply the number of times the function is triggered and used. It is essential to consider that an application with a higher number of fine-grained functions can lead to more invocations than an application with a smaller number of coarse-grained functions (Bortolini & Obelheiro, 2020). The resource consumption is the product of the allocated memory and execution time, and the CSP measures this in GB-s (Bortolini & Obelheiro, 2020). Calculators online support in determining the costs for the different FaaS services (Amazon Web Services, 2021).

The final matter to consider is performance. As shown in Figure 16, the execution times of functions are heavily dependent on a combination of provider, programming language, and memory allocation as



chosen by the organization (Bortolini & Obelheiro, 2020). Therefore, when execution times must be low, the used programming language and the amount of memory needed to run the functions determine what provider is the best fit.

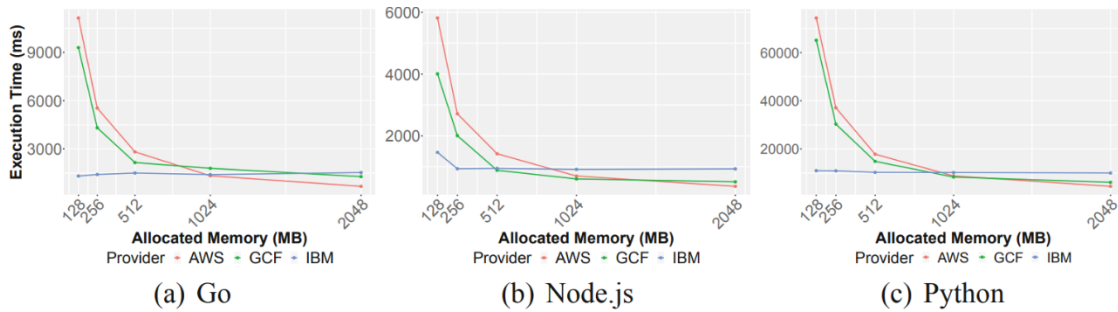


Figure 16: Performance differences of CSPs (Bortolini & Obelheiro, 2020)

## 2.4.3 Common patterns

The literature describes several common patterns that help develop Serverless applications and overcome challenges. This section discusses the most common.

*Externalised state:* Because functions must be stateless, a problem that developers often run into is that the state of a function might not be available in successive functions (Leitner, Wittern, Spillner, & Hummer, 2019). To retain the required states, the developers can store them in an external storage service. Realising external storage does induce extra latency and requires more programming effort (Leitner, Wittern, Spillner, & Hummer, 2019).

*Function Chain:* Functions hosted by FaaS providers are limited to a specific execution time. Those functions that require more time can be split up and chained together, multiplying the available execution time with the number of function parts (Leitner, Wittern, Spillner, & Hummer, 2019). Not every function can easily split up, causing extra development time to be incurred (Leitner, Wittern, Spillner, & Hummer, 2019). On top of that, calling synchronous functions triggers them twice, causing the CSP to bill double the costs (Nupponen & Taibi, 2020).

*Function Pinging:* Developers can occasionally ping functions heavily influenced by cold starts to keep them alive (Leitner, Wittern, Spillner, & Hummer, 2019). This way, the provider does not entirely drop the container of a function. Function pinging does induce costs for unused periods and contradicts the typical FaaS principles.

*Synchronous calls:* Because most API calls use a request/response format, architects prefer synchronous calls over asynchronous calls as they can cause complexity (Nupponen & Taibi, 2020). Therefore, using synchronous calls wherever possible is encouraged. Asynchronous calls are suitable for one-off jobs, such as triggering long-running backend processes; and within function chains (Nupponen & Taibi, 2020).

*Shared code:* Functions that share code might break when the code changes (Nupponen & Taibi, 2020). On top of that, functions might become too big due to shared code that is unnecessary for some functions (Nupponen & Taibi, 2020). Therefore, functions must be written independently and decoupled (Nupponen & Taibi, 2020).

*Libraries:* The package size of functions is limited to the constraints of the CSPs. Therefore, the number of libraries must be kept low and only those truly needed libraries may be imported (Nupponen & Taibi, 2020).

*Governance:* Adopting too many technologies, such as different languages and developing too many functions, can cause maintenance complexity (Nupponen & Taibi, 2020). This complexity requires higher skill levels for the people working on the project (Nupponen & Taibi, 2020). Therefore, the number of adopted technologies must be limited and governed by a team (Nupponen & Taibi, 2020). Architects must limit the number of functions by only developing necessary ones and grouping them into microservices. Grouping them must ensure that only the interface of the all-including microservice is exposed (Nupponen & Taibi, 2020).

## 2.5 Summary

This chapter presented Serverless's state of the art, including the definition, promise, suitability, and technology. Serverless is a serverless architectural style that enables organizations to develop applications without deployment and management responsibilities concerning the servers. The organization can outsource these responsibilities to a Cloud Service Provider. Serverless is the combination of two service types: FaaS and BaaS. When using FaaS, the organization breaks down its business logic into functions. Each function consists of a code written by one of the organization's developers and handed over to a CSP. The CSP is responsible for executing the code whenever it is required. BaaS services provide often-used functionality, such as storage and logging, provided entirely by the CSP.

Utilising Serverless leads to a reduction in an application's total cost of ownership since it lowers infrastructure, development, and maintenance costs. By shifting server management responsibilities towards the CSP, the developers can focus purely on developing the core business. It also realises higher utilisation of resources because the CSP only starts the functions when required. After execution, the CSP shuts down the function again to free the resources for other operations. This event-based operation reduces energy consumption and makes the organization more adaptive to change because the CSP scales the function on demand. Unexpected high workloads no longer result in a lousy performance, whereas low workloads do not result in unnecessary costs. The automatic scaling also makes the organization less prone to Denial-of-Service attacks, improving the application security. Serverless further improves security by removing instances after execution. The CSP destroys potential infections during removal, lowering the application's attack surface.

Not all applications are suitable for utilising Serverless because the CSP keeps parts of the application shut down until triggered. These functions must then boot up before they can execute the operation. This delay before execution is called a cold start and can significantly influence application performance. Functions that use interpreted languages such as JavaScript have shorter cold starts than compiled languages such as Java. Other influencing factors are the deployment package size, number of dependencies, and assigned memory and CPU. On top of the application characteristics, it is essential to consider the type of workload an application has, security implications, and vendor lock-in.

Serverless applications require a new architecture in which an architect splits up the functionality into triggerable isolated actions. Triggering happens through either package feeds or an API. A proven way of using Serverless is by utilising a microservice architecture. The architect first splits the application into

services and then divides them into smaller functions. The functions that deliver tailored needs run on FaaS services that execute the code provided by the organization; BaaS services can handle often used operations such as storage and logging. Choosing a CSP is crucial as they differ in services, billing prices, and performance. The number of function invocations determines the billed price, whereas the used programming language and the memory requirements influence performance. Many technical challenges exist among Serverless, but developers can use common practices that help overcome those.



# 3. Design & Development

## Realizing the Enterprise Serverless Assessment (ESA)



This chapter discusses the design and development of the artifactual solution within this research. The chapter starts by defining the organizational topics that we need to assess in Section 3.1. Sections 3.2 and 3.3 discuss the criteria within these topics according to literature and experts, respectively. Section 3.4 examines what criteria we must prioritize. The chapter ends with Section 3.5, which describes how we structure the criteria within a tool.

### 3.1 Assessment Topics

To ensure a comprehensive assessment, defining what areas within an organization are relevant during a Serverless adoption is essential. Because no framework regarding enterprise adoption concerning Serverless exists yet, as discussed in Section 1.2, we choose to base these topics on a framework concerning its parent concept: Cloud. For Cloud, many frameworks exist, providing the option to select the most appropriate one (Von Laszewski, Diaz, Wang, & Fox, 2012). Invent has developed a cloud strategy framework themselves, which claims to result in a "robust cloud strategy" that defines "the path which unifies the IT strategy with business goals" and "helps the organization optimize the existing IT landscape for maximized ROI" (Kumar, 2020). We deem this framework to be the best fitting one for this research because of the following reasons:

- This framework presents preliminary building blocks for a robust cloud strategy that can function as assessment topics for our artefact.
- Invent's consultants use these building blocks as guidelines in their internal communication and the communication with their customers when discussing Cloud topics. Using this framework will ensure both are aligned, and Invent can combine them at a later stage.
- The experts that are reachable for the expert opinion Delphi study are familiar with these topics.
- The organization willing to participate in the validation case study is familiar with these topics.

The Cloud Strategy Framework, shown in Figure 17, consists of eight building blocks. Invent's employees and customers often refer to these building blocks as topics. Therefore, the rest of this document also refers to them as (assessment) topics. Together, these topics define what we must consider for a comprehensive strategy. We shortly discuss each of them, starting at the top:

#### Maturity assessment

The maturity assessment topic is concerned with assessing digital capabilities and their current maturity. Here the prerequisites for the Cloud implementation are defined, which is in alignment with the goal of this research concerning Serverless. Therefore, we position the ESA as part of this Maturity topic. As prescribed within the framework, we assess the other seven topics to get the comprehensive assessment we require.

#### Cloud Target Picture

The Cloud Target Picture topic concerns the organisation's strategy when making decisions and defines what business goals justify Cloud adoption for a specific organization. This topic translates to the business goals that can motivate an organization for choosing Serverless. Consequently, we shorten the name of this topic to "Target Picture" for the rest of this research.

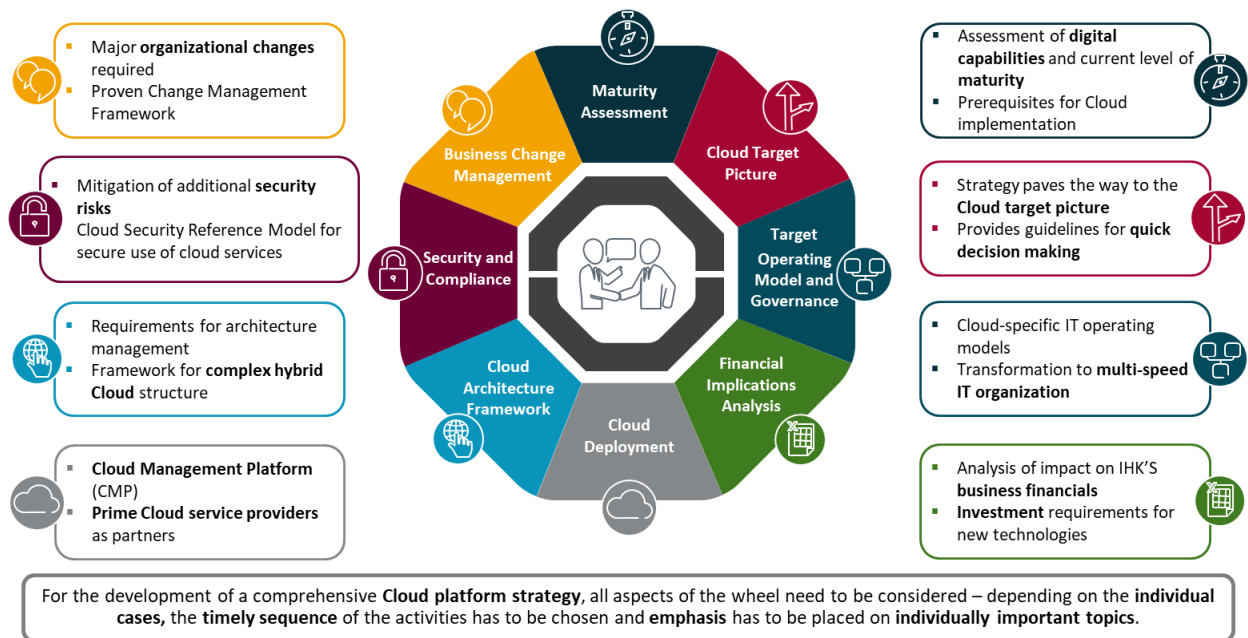


Figure 17: Capgemini Cloud Strategy Framework (Kumar, 2020)

### Target Operating Model and Governance

The Target Operating Model and Governance topic is concerned with choosing and implementing the correct IT operating models for running in the Cloud and the corresponding governance considerations.

### Financial Implications Analysis

The Financial Implications Analysis is concerned with all business financials during the adoption and successive operations. These financials include the organisation's required investments and the changes to the financial model.

### Business Change Management

The Business Change Management topic is concerned with the required organizational changes. An essential part of this topic is people change management.

### Security and Compliance

The Security and Compliance topic is concerned with risks and how to mitigate them. These risks can either be concerning security or compliance.

### Cloud Architecture Framework

The Architecture Framework is concerned with the requirements and changes to the architecture and how to manage them. These include all layers of architecture from enterprise to solution scale. We shorten the name of this topic to: "Architecture Framework" within the rest of this research.

### Cloud Deployment

The Cloud Deployment topic is concerned with the landing zone of the applications provided by the CSPs. Unlike the others, we do not shorten the name of this section as the Serverless applications deploy within a Cloud landscape, making the original name still applicable.

## 3.2 Literary Criteria

As described in Section 1.5, the first version of the ESA is a theoretical version based on the theory that is available in the literature. For this initial version, we include all criteria that fit one of the required topics and that we can derive from the Theoretical Framework presented in Chapter 2:

### Target Picture

The criteria for the Target Picture topic discuss the motives for using Serverless. Section 2.2 presents the promise of Serverless for an organization. When looking at this promise, we can distract the following motives:

- Improved elasticity and agility
- Lower total cost of ownership
- Faster time to market
- Reduced energy consumption
- Improved security

Because these motives do not translate into criteria but instead are goals that an organization can aim for or require, we verbalize these criteria as follows:

- *Does the organization aim for, or require, improved elasticity and agility for their applications?*
- *Does the organization aim for, or require, a lower total cost of ownership for their applications?*
- *Does the organization aim for, or require, a faster time to market for their applications?*
- *Does the organization aim for, or require, reduced energy consumption for their applications?*
- *Does the organization aim for, or require, improved security for their applications?*

### Target Operating Model and Governance

The criteria for the Target Operating Model and Governance topic:

- Business IT alignment
- Cloud operating model
- DevOps
- DevSecOps
- FinOps
- Technology governance
- Function governance

These aspects translate toward the following readiness criteria:

- *Are business and IT aligned within the organization?*
- *Does the organization deploy a Cloud operating model?*
- *Does the organization use a DevOps methodology?*
- *Does the organization use a DevSecOps methodology?*
- *Does the organization use a FinOps methodology?*
- *Is capacity available for increased technology governance?*
- *Is capacity available for function governance?*

## Financial Implications Analysis

The criteria concerning financials and investments:

- Unintended high infrastructure costs

This aspect translates toward the following criteria:

- *Are financial buffers available for unintended high infrastructure costs?*

## Business Change Management

The criteria concerning organizational changes:

- Familiarity regarding Serverless advantages
- Experienced with Functional Programming
- Experienced with the Immutable Infrastructure Paradigm
- Experienced with the Common Serverless Patterns
- Experienced with Interpreted Programming Languages
- Responsibility for financial performance

These aspects translate toward the following criteria:

- *Are relevant stakeholders aware of the Serverless advantages?*
- *Are developers experienced with Functional Programming?*
- *Are developers experienced with the Immutable Infrastructure Paradigm?*
- *Are developers experienced with the Common Serverless Patterns?*
- *Are developers experienced with Interpreted Programming Languages?*
- *Do developers feel responsible for the financial performance of their applications?*

## Security and Compliance

The criteria concerning the risks that Serverless causes to security and compliance:

- Validating and securing communications

This aspect translates to the following criteria:

- *Are developers able to validate and encrypt all communications?*

## Architecture Framework

The criteria concerning architecture:

- Service-Oriented Architectures/Microservices
- Reuse of functionality
- Varying execution times can lead to performance issues.

This aspect translates to the following criteria:

- *Does the organization have or plan applications with service-oriented architectures?*
- *Does the organization emphasize reusing functionality?*
- *Does the organization have applications with predictable execution times?*



## Cloud Deployment

The aspects related to Cloud Deployment:

- Vendor lock-in risk
- CSP Partnerships
- Database, API gateway, logging, and IaaS Services
- Supported runtimes/programming languages
- Testing and CI/CD tools

These translate to the following criteria:

- *Can the organization accept a more severe vendor lock-in?*
- *Does the organization have partnerships with a CSP?*
- *Does the organization use database, API gateway, logging, and IaaS services provided by a CSP?*
- *Does the organization develop with programming languages that a CSP supports?*
- *Does the organization have testing and CI/CD tools in place?*

## 3.3 Expert Criteria

As described in Section 1.5, the theoretical version of the ESA is to be criticized and complemented by experts who have experience with Serverless adoptions. To gather these experts' opinions, we perform a Delphi study consisting of two rounds:

### 3.3.1 Delphi Round 1

During round 1, the experts can openly present what they believe is relevant to each topic. Hereafter, they criticize the findings in the literature. This section discusses the results of round 1, including the added parts, modifications, and removals. Appendix 1: Delphi contains all contribution details.

#### Target Picture

##### **Added**

*High availability for applications:* This is a significant benefit that does not necessarily have to be part of increased agility and elasticity. It can be a motive reason on its own.

*Enables future innovations such as the Internet of Things and Edge Computing:* Serverless is a good fit with these types of innovations. Transforming your applications to use Serverless ensures they can better adopt those in the future.

*More efficient use of resources, leaner:* Serverless ensures that developers can spend less time on the operations of an application and focus purely on developing the core business logic. This reduction enables the organization to realize more with fewer people.

##### **Modified**

*Improved agility and elasticity:* This motive first included high availability, which the experts deemed a stand-alone motive. We choose to modify this motive accordingly.

*Lower total cost of ownership:* The original infrastructure can only be scaled down after an extended period, causing double fees initially. Therefore, as the prices will drop in the long run, the application becomes cost-effective after the investment period. We modified this motive accordingly.

### **Removed**

*Reduced energy consumption:* While Cloud does reduce energy consumption, this is not necessarily the case for Serverless. Serverless applications cause more overhead that might increase energy consumption again. Therefore, the experts state this is not a justifiable motive.

*Improved security:* While Serverless brings some security benefits, new risks also arise. Serverless applications are, therefore, not by definition more secure. Thus, the experts state this is not a justifiable motive.

## **Target Operating Model and Governance**

### **Added**

*Distributed, non-silo, organization:* Siloed organizations separate responsibilities over different departments. Serverless technology influences development, operations, financials, and security. By breaking silos, these departments become one enabling faster decision making and removing any internal competition.

*BizDevSecOps – End-to-End delivery teams:* The final evolution of DevOps, here the business becomes part of the development, which speeds up decision making and truly enables the faster time to market that Serverless promises.

*Open standards:* To counter the more severe vendor lock-in that comes with Serverless, the organization must strive for using open standards that multiple CSPs support. Using open standards enables the organization to switch more manageable.

*Event-driven architectures:* Serverless applications use event-driven architectures. Expertise regarding these architectures is scarce, and acquiring it requires effort.

*Responsibility at lower layers in the organization:* To benefit from the faster time to market that Serverless promises, the organization's operating model must allow this by ensuring that teams can make decisions fast. The responsibility must lie low within the organization to realise this.

*Platform-based approach:* A platform-based approach ensures faster and easier delivery by using templates to determine how and where to position applications quickly. This approach fits Serverless because of the unique services that each CSP offers.

### **Modified**

*Function Governance → Information Governance:* Function governance results in less autonomous teams, which conflicts with the desired situation of end-to-end delivery teams. The organization must, however, monitor what team manages what information. This need is due to the potential risk of leaking business secrets, such as algorithms implemented within a FaaS function.

## Removed

*Business IT Alignment:* The experts did not deem this criterion relevant to an organisation's Serverless readiness. The organisation's business side is not interested in the technology side of an application. Therefore, the experts state this is not a justifiable criterion.

*Technology governance:* Technology governance results in less autonomous teams and slows decision making. This criterion conflicts with the desired end-to-end delivery teams. Therefore, the experts state this is not a justifiable criterion.

## Financial Implications Analysis

### Added

*Large investment - Costs will only drop after a long period:* When migrating an application to Serverless, the original application will have to run on the original infrastructure until it is ready. This double infrastructure means that you will pay for both at the beginning of the migration. The costs will drop only after a long period.

*Costs difficult to assess:* Costs in Serverless are relatively unpredictable compared to other Cloud offerings due to the pay-per-use billing model. This unpredictability means that large cost fluctuations can occur.

*Different billing model:* The pay-per-use billing model used in Serverless differs from typical Cloud services that use subscription-based billing models. This change in the billing model impacts the organization's financial model.

*Increased training and recruitment costs:* The organization must obtain the knowledge required to develop and run Serverless applications through extra training and recruitment. These extras result in increased costs for which the organization must make a budget available.

*Increased security costs:* Security becomes more complicated when using Serverless due to a lack of tooling. This lack requires the employees to perform more manual actions, leading to an increase in security costs.

*Increased governance costs:* Serverless requires more governance, leading to increased work that needs to be done by employees. This extra work increases the governance costs.

### Modified

*Unintended infrastructure costs:* These costs rarely happen in practice. Only inexperienced developers sometimes make mistakes that lead to unintended infrastructure costs. By ensuring these people work with quotas, we can mitigate this risk. Hence, we modify this criterion accordingly: Are quotas placed on the infrastructure capacity during development, especially in the case of inexperienced developers?

## Business Change Management

### Added

*Agile teams:* While waterfall methods work when developing Serverless applications, it is better to work with agile teams. Agile teams deliver faster and are more flexible, which aligns better with the advantages of Serverless.

*Purpose:* Development teams need to have a sense of purpose when working on projects. They must be committed to the organization and the product they work on.

*Business teams with IT responsibility:* Using Serverless technology will increase IT responsibility among business teams. These teams usually feel uncomfortable with this responsibility and require guidance.

*Innovative projects close to the business:* Serverless works best in innovative projects and is closely involved with the business teams of the organization.

*Included in all strategies:* Teams within the organization must know that Serverless is an option within their toolbox. They need guidelines on how and when to use it by adopting it in all strategies: Cloud, platform, and architecture

*Experienced with a Cloud-Native mindset:* Development teams need to have experience with the required methodologies to build in a cloud-native way. The cloud-native mindset comprehends these ways of working.

*Experienced with Separation of Concerns:* This design principle is essential for separating computer programs into distinct sections based on the concerns they address. Knowledge about this principle is necessary within the development teams.

### Modified

*Familiarity regarding Serverless advantages:* Apart from the benefits, it is also essential that people know the disadvantages of using Serverless. All parties involved with the IT process need to be aware of both. This criterion is therefore modified accordingly.

*Experienced with the Immutable Infrastructure Paradigm:* Not every developer needs this experience, but the team requires some expertise. This criterion is therefore modified accordingly.

*Experienced with the Common Serverless Patterns:* Not every developer needs this experience, but the team requires some expertise. This criterion is therefore modified accordingly.

*Responsibility for financial performance:* The individual developers may not be held responsible for the financial performance of the applications. They need to have insights into the application to make changes when necessary. The responsibility lies with the entire development team. This criterion is therefore modified accordingly.

### Removed

*Experienced with Interpreted Programming Languages:* Interpreted programming language knowledge was deemed broadly available and easy to obtain, so not relevant for Serverless readiness. We, therefore, removed this criterion.

*Experienced with Functional Programming:* Functional programming knowledge was deemed easy to obtain and not a determining factor for a Serverless adoption. We, therefore, removed this criterion.

## Security and Compliance

### Added

*Expertise department:* It is essential to realize an expertise department to which developers can reach out if they require support. The essence of this department is not to centralize responsibility but to centralize knowledge.

*Difficult to monitor and audit:* Because the number of components is higher in Serverless applications, and these components can completely scale down, it is more challenging to monitor and audit them as trails are more difficult to trace.

*Complex security by design:* Experts expect that performing security by design method is more difficult for Serverless applications.

*Code-based compliancy and security checks:* Because Serverless components can completely scale down, it is essential that compliance and security checks are performed on code base directly within the Ci/Cd pipeline.

*Risks regarding public interfaces:* Developers must become aware of the risks involved with public interfaces because all parts within a Serverless application can communicate through open channels.

*Policies for new components in the IT landscape:* It is essential that teams remain autonomous but only add components to the IT landscape that are secure. Guidelines for adding new features that enforce security measures are therefore required.

*Shielding components with networks:* The development teams must shield the Serverless components with a network to prevent unauthorized access.

### Modified

*Validating and securing communications:* The individual developers cannot oversee the complete landscape, so they cannot be responsible for validating and encrypting all communications. The responsibility must lie with the entire development team. This criterion is therefore modified accordingly.

## Architecture Framework

### Added

*Well architected framework:* Experienced architects publish frameworks that demonstrate proven architectures. The enterprise's architects must follow these frameworks to quickly and adequately design new solutions without inventing everything themselves.

*Serverless common/best practices:* Architects must know the Serverless common/best practices to deal with often faced challenges. These practices can be learned through training and used within well-architected frameworks.

*Event-driven architectures:* Serverless applications operate through event-driven architectures. Knowledge about these architectures is scarce, and acquiring it is crucial for developing Serverless applications.

*Streamline Serverless definition:* No formal description for Serverless exists. Each person can mean other technologies or services when talking about Serverless. It is vital to streamline a definition to prevent miscommunication within the organisation.

*Low function points:* Architects must prevent complexity by keeping the function points in parts of the application low. Complexity in the application can cause teams to lose their autonomy or make processes within the operating model longer.

*Vendor lock-in:* Moved from the Cloud deployment topic as this lock-in is mainly happening at the architectural level.

### **Modified**

*Service-Oriented Architectures/Microservices:* Serverless works best with modular or distributed architectures. These modules do not have to be service-oriented but can also be technology-oriented. This criterion is therefore modified accordingly.

*Emphasize reuse of functionality:* The organization should not emphasize reuse but only facilitate it. This criterion is therefore modified accordingly.

*Varying execution times can lead to performance issues:* Experts deem the execution times irrelevant for Serverless readiness. Architects should have performance insights for all the applications to ensure they know what parts to migrate. This criterion is therefore modified accordingly.

### **Cloud Deployment**

#### **Added**

*Infrastructure as Code and Immutable Infrastructure:* Developers may no longer manually change any deployment settings. They must define everything within Infrastructure as Code.

*Central logging database:* Because logging becomes more complex when using Serverless, realising a primary solution to log applications is crucial.

#### **Modified**

*Database, API gateway, logging, and IaaS services:* IaaS services are not necessary when building Serverless applications. Monitoring services are. We, therefore, modified this criterion accordingly.

*Testing and CI/CD tools:* On top of these, an organization needs to have version control tools that fit Serverless development. We, therefore, modified this criterion accordingly.

#### **Removed**

*Vendor lock-in:* The vendor lock-in was deemed more relevant within the Architecture Framework topic as the lock-in happens at the architectural level. We, therefore, removed this criterion from this topic.

*Supported runtimes/programming languages:* The difference in runtimes between the CSPs exists, but experts do not deem them relevant for Serverless readiness. We, therefore, removed this criterion.

*CSP partnerships:* Partnerships can lead to lower costs for an enterprise, but experts do not deem them a relevant prerequisite for Serverless adoption. We, therefore, removed this criterion.

## 3.3.2 Delphi Round 2

During the second round, we present an updated version of the ESA to the expert panel. This updated version includes the changes from round 1, and we ask the experts to verify the changes and propose further improvements. This section discusses the changes resulting from this second round.

### Target Picture

#### Modified

*Lower total cost of ownership in the long run:* Only migrated applications suffer from initial double infrastructure costs. New build applications directly benefit from more cost-effective infrastructures. This cost-effectiveness only improves compared to other Cloud services, not necessarily to on-premise solutions. This motive is therefore modified accordingly.

*More efficient use of resources, leaner:* Experts state that we should not refer to employees as resources. We, therefore, rephrase this motive to more efficient use of (human) resources.

### Target Operating Model and Governance

#### Modified

*Distributed, non-silo, organization:* A distributed organization is not necessarily a non-silo organization. We, therefore, rephrase this criterion accordingly.

*BizDevSecOps – End-to-End delivery teams:* These teams are not always known as BizDevSecOps, but people in the industry also refer to them as Fusion Teams or Product Oriented Delivery teams. We, therefore, modify this criterion accordingly.

### Financial Implications Analysis

#### Modified

*Unintended high infrastructure costs among inexperienced developers:* This criterion should not focus on the individual developers but on their teams. We, therefore, modify this criterion accordingly.

*Large investment - Costs will only drop after a long period:* The initial investment is not necessarily high. The critical part is the costs that only fall after a long period. We, therefore, modify this criterion accordingly.

#### Removed

*Increased security costs:* The security costs should no longer increase as new Serverless security tools have become available. We, therefore, removed this criterion.

### Business Change Management

#### Removed

*Purpose:* While purpose is good for the effectiveness of a development team, experts do not deem it necessary for Serverless adoption. We, therefore, removed this criterion.

*Innovative projects close to the business:* While Serverless is more likely to be used in innovative projects, they do not necessarily have to be close to the business teams. Even without business involvement, the development teams can create Serverless applications. We, therefore, removed this criterion.

## Security and Compliance

### Modified

*Difficult to monitor and audit:* It becomes essential to automate auditing processes due to the increase in components caused by Serverless. We modified this criterion accordingly.

### Removed

*Security by design gets more complex:* This increased complexity is not the case, according to the experts. We, therefore, removed this criterion.

*Shielding components with networks:* Networks counteract the idea of how Serverless applications interact. We, therefore, removed this criterion.

## Architecture Framework

### Modified

*Streamline Serverless definition:* The organization does not have to streamline the definition but must formalize it. They need to define what technologies and services to include and exclude. We modified this criterion accordingly.

*Low function points:* Some experts deem function points to be an outdated phenomenon. Complexity must remain low by preventing accumulations of different components and services. We modified this criterion accordingly.

*Serverless common/best practices:* The common/best patterns are also important. We modified this criterion accordingly.

## Cloud Deployment

### Modified

*Database, API gateway, logging, and monitoring services:* On top of these, the CSP should also provide security services. We modified this criterion accordingly.

*Central logging database:* Centralizing logging is troublesome and causes teams to lose autonomy. However, you need a distributed monitoring solution to monitor the different loggings databases. We modified this criterion accordingly.

## 3.3.3 Delphi Conclusion

As discussed in Section 1.5, now the results of the two Delphi rounds are in, it is time to review if the Delphi expert panel came to a consensus. Looking at the low number of changes in round 2, presented in Section 3.3.2, and the magnitude of these changes, which are only minor modifications instead of complete additions or removals, we can state that they reached a consensus. This consensus implies that the criteria, including the changes after round 2, are final and included in the ESA. A complete overview of these criteria is available in Appendix 2: Delphi Consensus.



### 3.4 Prioritization

Because not all topics or criteria might be equally important, we aim to distinguish them. By doing so, the consultant can directly inform the organization on what actions to prioritize. We base this prioritization on two factors: impact and effort. In the ESA’s context, impact means the extent to which a criterion improves an organization’s readiness for Serverless adoption. Effort means the amount of work the organization must carry out to meet this criterion. Using these two factors, we can assign prioritization scores for each action. Figure 18 shows how the translation of these factors to their respective priorities.

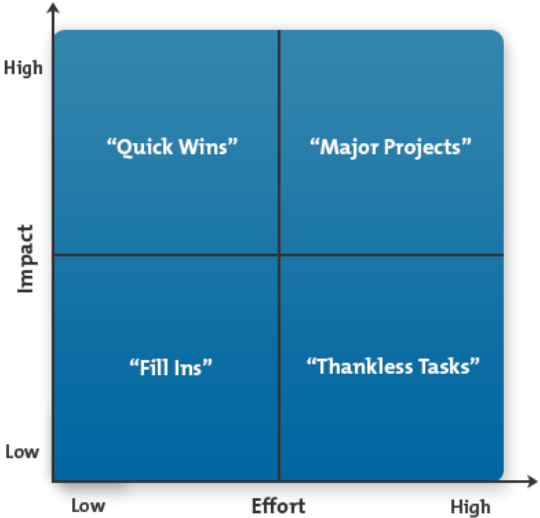


Figure 18: Action Priority Matrix (MindTools, 2022)

To determine what criteria have the most impact and require the most effort, we appeal to the Delphi expert panel because of their hands-on experience. We ask what criteria they believe are most impactful and laborious for each topic. Whenever two or more experts pick a criterion for one of the two factors, this respective factor is determined to be “high”. When a single expert picks a criterion, this factor is determined to be “medium”. When none of the experts chooses a specific criterion, this criterion is determined to be “low” for both factors. The prioritization is done based on the scores in Table 4. A lower score implies a higher priority. Appendix 3: Prioritization Details shows the details for each criteria priority.

Impact	High	1	2	3
	Medium	2	3	4
	Low	3	4	5
		Low	Medium	High
		Effort		

Table 4: Action Prioritization Scores

In the Target Picture topic, the experts stated that no motive for using Serverless transcends the others. They claim that each of the found motives is satisfactory when choosing to use Serverless as long as it aligns with the organization's strategy. On top of that, the experts state that strategy must always lead to technology, never the other way around. We, therefore, choose not to assign impact or effort scores to this topic.

## 3.5 Assessment Tool

To ensure every consultant can assess their respective organization, we develop a tool that structures the ESA. The tool consists of three main parts: a questionnaire, a results dashboard, and a plan of action. Apart from these, the tool contains:

- Front-page
- Contents overview
- Instructions on the scope of the assessment, the role of the consultant, who to interview, and the assessment topics
- Future steps
- Epilogue
- Settings page

We developed the tool within Microsoft Excel because it is widely available within organizations. Most people know how to use Excel, making the ESA easy to use and improve. On top of that, organizations do not need to pay extra license fees when using the Excel worksheets apart from the software itself. Appendix 4: ESA Tool shows a complete overview of the tool.

### 3.5.1 Questionnaire

The first step is for the consultant to determine if the organization meets the found criteria. Therefore, the initial part of the ESA is a questionnaire through which the consultant gathers the required information. The consultant might already have this information or can accumulate it by interviewing employees within the organization or performing additional research. Figure 19 shows this part of the tool with some of the criteria.

### 3.5.2 Results

The ESA tool must determine and present the results through the questionnaire's answers. These results need to inform the organization whether or not Serverless is the right innovation for them from a strategic and readiness point of view. Therefore, the ESA tool calculates two scores: "Strategic Fit" and "Organizational Readiness". Strategic fit is the degree to which an opportunity matches the organization's strategy. Looking at the assessment topics presented in Section 3.1, the Target Picture topic aligns best with this score due to its focus on motives for using Serverless. Organizational readiness is the extent to which an organization is prepared to implement organizational change. The remaining six topics align most with this score due to their focus on the organization's capabilities. We express both scores as percentages, where 0% implies the organization meets no motives or criteria and 100% means it meets all motives or criteria. The tool bases the strategic fit score on a square root function due to the claim of the experts that one motive is enough for using Serverless. The score, therefore, jumps directly to the

indicative minimum of 50%. The organizational readiness score is linear, but the criteria with a higher impact weigh heavier in the calculation. The tool combines both scores into either fit, potential, or unfit advice. Fit implies that at least one motive for using Serverless aligns with the organization's strategy, and the organization meets at least half of the criteria. Potential means that at least one motive aligns but the organization meets less than half the criteria. Unfit indicates no motive for using Serverless that aligns with the organization.

The results page, shown in Figure 20, presents the individual fit and readiness scores, the advice, and the scores of each separate topic. On top of that, a spider graph shows how this organization compares to the industry standard. The tool determines this standard based on the average scores from all previous assessments. Finally, the tool shows the organization's current position within a matrix with both scores at the axis. This matrix distinguishes four zones: Limited value & readiness, valuable investment, hidden gem, and high value & readiness. When the organization is within the limited value & readiness zone, it will neither benefit from Serverless nor will the adoption be smooth. Looking at other investments is then the best option. The valuable investment zone implies that Serverless will be a good investment but needs some work for a smooth adoption. The hidden gem zone suggests that the organization can smoothly adopt Serverless but will not benefit from it right now. Serverless is then a hidden gem because it can become a valuable innovation when the organization chooses to anticipate the benefits. High value & readiness implies the organization can start the adoption and will experience the benefits.

### 3.5.3 Plan of Action

The last central part of the ESA is the plan of action. This section presents steps that can improve the organization's readiness. The action plan suggests an action for each criterion and their prioritization score as described in Section 3.4. What steps to perform is dependent on the organization. The consultant is encouraged to choose the actions they believe are most fitting based on their knowledge about the organization. Figure 21 shows this part of the tool with some suggested steps.



Previous step: Overview

Next step: Results



### Target Picture

*(Select criteria cell for additional information)*

*Guidelines for quick decision making*

*(Comments cells are available to fill in additional information about the answer)*

Criteria	Comments	Response	Impact
Does the organization aim for, or require, a faster time to market for their applications?		FALSE	N/a
Does the organization aim for, or require, improved elasticity and agility?		FALSE	N/a
- Do one or more applications have unpredictable workloads?		-	-
- Do one or more applications have significant idle time?		-	-
Does the organization aim for, or require, a lower TCO for their applications?		FALSE	N/a
- Do development teams spend, on average, more than 25% of their time on maintenance tasks?		-	-
- Does pre-planning applications during development require a significant part of the available capacity?		-	-
- Do applications have a cost-ineffective infrastructure? I.e. a lot of idle time or significant fluctuations in workloads?		-	-
Does the organization aim for, or require, increased application availability?		FALSE	N/a
Does the organization aim for, or require, enabling future IT innovations? I.e. IoT or Edge		FALSE	N/a
Does the organization aim for, or requires, more efficient use of (human) resources?		FALSE	N/a

### Architecture Framework

*Requirements for architecture management*

**Readiness Impact**

**High**

Criteria	Comments	Response	Impact
Does the organization have/plan applications with modular/distributed architectures?		FALSE	High
Does the organization facilitate re-use of functionality/code/services?		FALSE	High

Figure 19: Questionnaire

# Enterprise Serverless Assessment

Results

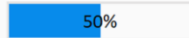


Previous step: Questionnaire

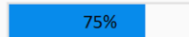
Next step: Plan of Action



Your Strategic Fit score is:



Your Organizational Readiness score is:



**Fit**

This score implies that the business is a good fit for Serverless Technology and ready to utilize it. Using the advice for the individual topics can still benefit the adoption process.

Score	Area	Impact	Score
Strategic Fit	Target Picture	N/a	50%
Organizational Readiness	Architecture Framework	High	73%
	Target Operating Model & Governance	Top	68%
	Business Change Management	Top	74%
	Security and Compliance	High	100%
	Financial Implications Analysis	Basic	43%
	Cloud Deployment	Basic	100%

Suitability Spider Graph

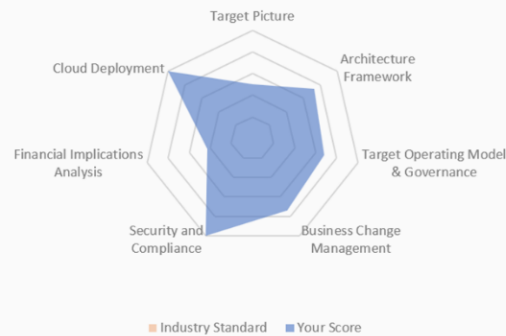


Figure 20: Results



Previous step: Results

Next step: Future Steps



### Target Picture

*Guidelines for quick decision making*

**There are no actions defined for the Target Picture topic.**

x

Because this section focusses purely on strategy, no actions for this section are defined. Strategy should always lead technology, not the other way around. If the strategy fit is insufficient, consider looking at other technologies. Whenever the organization does have a high readiness score, Serverless might be a hidden gem that can deliver value fast if the organization choses to capitalize on this opportunity.

### Architecture Framework

*Requirements for architecture management*

Score

0%

Readiness Impact

High

#	Advice	Impact	Effort	Priority
-	Modular / distributed architectures such as service-oriented architectures (SOA) or microservices, have been proven to be a good fit for Serverless. Migrating from a Monolithic application towards such an architecture can be costly and time consuming but does come with a lot of benefits and better Serverless adoption. Check if such a transformation might be a valuable investment.	Medium	High	4
-	Serverless enables reusing code and functionality within (and across different) applications. Reuse can save developers a lot of time compared to doing everything by themselves. This can be facilitated by the organization through a code sharing platform. Consider informing developers about the benefits of re-using and realize/adopt a platform where code can be shared.	Medium	Low	2
-	No singular definition for Serverless exists. This means that it is important to align a definition of Serverless within the organization to prevent miscommunication. Define what services offered by the CSP are part of the Serverless offerings used by the organization.	Medium	Low	2
-	As claimed by AWS: "AWS Well-Architected helps cloud architects build secure, high-performing, resilient, and efficient infrastructure for a variety of applications and workloads.". Using such an architecture ensures the organization uses the best practises according to the CSP without having to figure out everything themselves. Consider following the well-architected framework of the organization's CSP for easier adoption.	High	Low	1
-	Designing Serverless architectures can best be done by using common/best practises. Ensure the organization's architects are aware of these by having them follow a course of training.	Medium	Low	2
-	Serverless services are event-driven, this means that the organization's architects must have experience with event-driven architectures. This experience is scarce and extra training might be required. Consider having them follow a course or extra training	Medium	Medium	3
-	Serverless works best for applications with invariable execution times because variable execution times can lead to performance issues. Ensure that development teams have technical insights on the applications to prevent these issues by doing performance testing on them.	Low	Low	3
-	Complexity within applications endangers the autonomy of teams. Keeping application complexity low ensures teams stay independent. Ensure that architects try to prevent accumulation of components.	Low	Medium	4

Figure 21: Plan of Action

# 4. Demonstration



## Performing the ESA within conditions of practice

This chapter describes the demonstration of the ESA at an organization to validate if it works as intended. As described in Section 1.5, we perform this demonstration by assessing a multinational financial service provider during two sessions. We first fill in the questionnaire with an architect from this organization which we discuss in Section 4.1. We present the corresponding results and advice in Section 4.2.

### 4.1 Questionnaire

During the first sessions, we informed the organization’s representative about the aim of the ESA, the assessment topics, and the assessment methodology. After that, we asked and got permission to record the session and started with the questionnaire. We asked the representative a question for each of the criteria. After each answer, we directly filled in the response within the tool. We used the comment fields when the representative provided additional information. Table 5 shows the filled in Architecture Framework part. The full details of the questionnaire are available in Appendix 5: Demonstration Details.

Criteria	Comments	Response
Does the organization have/plan applications with modular/distributed architectures?	They are working hard to realize these architectures.	TRUE
Does the organization facilitate the re-use of functionality/code/services?	They are working hard to become API - driven: an essential part of their agenda	TRUE
Is there alignment within the organization regarding the scope of Serverless? I.e. what service offerings are included, and how are they used?	It is not defined, but an informal definition causes little discussion: They state it is the technologies where you do not run the instance yourself. Part of PaaS/FaaS/BaaS	FALSE
Do (solution) architects use their CSP's well-architected framework?	They just started to do so. This process comes up to speed.	TRUE
Do (solution) architects know the Serverless common/best practices?	Working on getting there, but a long way to go. On a scale of 1-5: 2	FALSE
Do (solution) architects have experience with event-driven architectures?	Working on getting there, but a long way to go. On a scale of 1-5: 2	FALSE
Do (solution) architects have insights into the performance of the applications? For example, through performance tests?	Competence centre is present - Regarding critical apps, they know everything	TRUE
Do (solution) architects keep application complexity low? I.e. Microservices over significant accumulations of functionality?	They are starting to get there. On a scale of 1-5: 3	TRUE

Table 5: Architecture Framework questionnaire part during demonstration

The session took about 50 minutes, of which 10 minutes were the introduction and 40 minutes was filling in the questionnaire. The representative was able to answer all questions but did sometimes use a scale instead of true or false. These answers required the consultant to decide if they met the criterion.

## 4.2 Results

With the answers to the questionnaire filled into the tool, the assessment results are directly available. The scores, shown in Table 6, state that the organization is fit for using Serverless: They align with a single motive for using the technology and meet at least 50% of the criteria.

**Your Strategic Fit score is:** 50%

**Your Organizational Readiness score is:** 71%

**Fit** This score implies that the business is a good fit for Serverless Technology and ready to use it. Using the advice for the individual topics can still benefit the adoption process.

Score	Area	Impact	Score
Strategic Fit	Target Picture	N/a	50%
Organizational Readiness	Architecture Framework	High	60%
	Target Operating Model & Governance	Top	68%
	Business Change Management	Top	74%
	Security and Compliance	High	83%
	Financial Implications Analysis	Basic	57%
	Cloud Deployment	Basic	100%

Table 6: Demonstration Results

As shown in Table 6 and visualized in Figure 22, the organization meets at least 50% in all the areas. Because this is the first assessed organization, no industry standard to compare it is available yet.

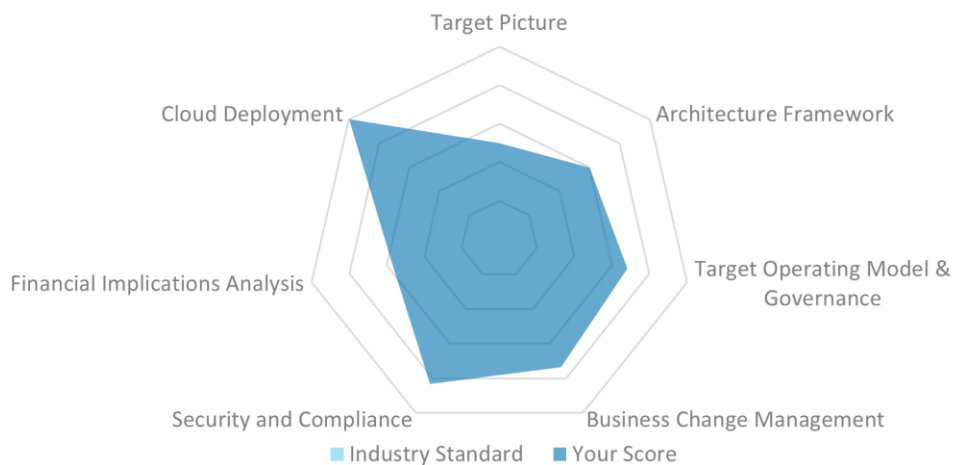


Figure 22: Demonstration Spider Graph



The final part of the results is the current position matrix shown in Figure 23. This figure shows that the assessed organization is currently up against the high value & readiness zone. This zone implies that Serverless is a valuable innovation for this organization, and they are ready to adopt it. The organization can still improve their readiness and obtain more value if it anticipates more Serverless benefits.



Figure 23: Demonstration Position Matrix

### 4.3 Advice

The questionnaire and results lead to several pieces of advice. We based the advice on the results in the tool combined with the knowledge we have about the organization. During the second session, we presented it. This session took about 1 hour, of which we spent 30 minutes on communicating the advice:

#### Architecture Framework – Score: 60%

Strong capabilities: Aiming for modular architectures; Facilitating code re-use; Using well-architected frameworks; Development teams have financial insights;

Because the organization only has an informal definition for Serverless:

- Define an architectural definition for Serverless within the organization to prevent miscommunication. Proposal: Define what CSP services the organizations include and exclude.

Because the knowledge about Serverless best/common practices and event-driven architectures is not widely available among the organization’s architects:

- Remember that architects with knowledge about Serverless best/common practices and event-driven architectures are scarce. Proposal: Try to ensure that this knowledge is centralized and made available in a context that matches the organization. This way, it is easier for architects without this knowledge to obtain it.

### Target Operating Model & Governance – Score: 68%

Strong capabilities: Following a cloud operating model; DevOps/FinOps/End-to-End delivery teams; Using a platform-based approach; Performing information governance;

Because security by design/DevSecOps in practice does not always hold within the organization:

- Try to enforce security by design through Ci/Cd pipelines or increased governance upon the solution architects.

Because the organization does not support multi-vendor standards against vendor lock-in but only focuses on migrating business logic:

- Keep in mind that not all CSPs support the same runtimes. Ensure that multiple vendors at least support the programming languages used by the development teams.

Because the organization is still working hard to break down organizational silos:

- Ensure that you know how the framework used to break down these silos might impact the current Cloud/Serverless architecture.

### Business Change Management – Score: 74%

Strong capabilities: Serverless adopted in all strategies; Agile teams; Experienced developers

Because some parties are uninvolved while communicating the choice for Serverless:

- Ensure all parties that are involved with the IT process are aware of the advantages and disadvantages of Serverless

Because not all developers have the required knowledge and skills:

- Gather the progressive developers for the Serverless projects, and promote knowledge sharing and knowledge communities/guilds

### Security and Compliance – Score: 83%

Strong capabilities: Security expertise department present within the organization; Developers able to secure and validate communications; Awareness about the risks of public interfaces; Policies for the IT landscape; Code based compliance and security checks in the Ci/Cd pipeline; Automated alerting;

Because automated recovery is limited to alerting:

- The increase in components when using Serverless makes manual healing tasks unsustainable. Ensure that critical applications get automated healing solutions.

### Financial Implications Analysis – Score: 57%

Strong capabilities: The organization is aware of the long term investment requirements; Organization can deal with unpredictable costs; Development teams have infrastructure quota's against extreme traffic costs;

Because the billing model will change from monthly subscription costs to pay-per-use:

- Inform the financial department about changes to the financial model and prevent issues by creating awareness and renovating the financial processes appropriately.

Because the costs of training, recruitment, and governance can increase when further adopting Serverless:

- Request budget for increased training and recruitment costs
- Request budget for increased governance costs

#### Cloud Deployment – Score: 100%

Strong capabilities: Possesses and uses the right services and tools; Enforcing Infrastructure as Code; Multiple distributed monitoring solutions in place;

Because the organization meets all criteria within this topic, no further advice is necessary.



# 5. Evaluation



## Determining the effectiveness of the ESA in guiding organizations

This chapter evaluates how well the ESA solves the research problem. In other words, does the ESA support an enterprise organization with their Serverless adoption by providing the necessary guidance? As described in Section 1.4, we evaluate the ESA on two aspects: usability and effectivity. We discuss these aspects in Section 5.1. Section 5.2 discusses the limitations of the ESA.

## 5.1 Usability & Effectivity

This section evaluates the assessment process by discussing how the organization experienced the sessions and results. This discussion includes the methodology, time, assessment topics, criteria, prior expectations, self-perception, results, and advice. We first present the expert's opinion, after which we discuss the corresponding improvements we made.

### 5.1.1 Expert opinion

Because a large scale adoption can take years, it is impossible to examine and measure the long-term effectiveness of the ESA on an actual adoption process. Fortunately, as described in Section 1.5, the organization that participated during the demonstration already has experience with Serverless and invested a lot in its adoption. By asking for feedback from the representative, we can see if the ESA's results align with their experiences and if they think it would have benefitted their adoption process if they had the ESA's insights at their disposal when they started the adoption.

#### Methodology

The organization received the ESA's methodology well: The initial question session, followed by the results and advice session, felt intuitive. However, we asked the questions in an alternative response way during the first session: Yes or no. They had expected a scoring method with a scale: Score the organization on criteria X where one is the lowest and five the highest. A scoring scale felt more appropriate for the representative because some employees might be more optimistic than others. This bias would be more apparent when using a scoring system. This need also became clear during the demonstration as the representative gave some answers with a score: "Yes, we do, but on a scale from 1 to 5, we are only at 2."

#### Time requirements

The time required to perform the ESA was worth the insights it provided. It took about two hours which is much shorter than it would have taken to research everything yourself, especially when an organization is beginning its adoption process. The expectation is that employees from strategical layers would value the ESA more than those in operational layers, which usually are more inclined to investigate things themselves.

#### Assessment topics

The assessment topics felt relevant and comprehensive for all parts of the adoption process. The ESA discussed each topic well, so no subjects were underexposed. The criteria felt appropriate in their respective topics.

## Criteria

The criteria felt well thought out and relevant when reflecting on our Serverless adoption. The following criteria need improvement according to the organization:

- Parties involved with the IT process need to be aware of the advantages and disadvantages of Serverless: Our management teams are not interested in technology affairs; within our organization, this would not work
- Extra budget is required for increased governance costs: These costs should not increase if done correctly. We did not see any increase within our organization
- The organization must deal with unpredictable infrastructure costs: At our scale, these costs are only unforeseen during the first year

## Prior expectation

The ESA is in line with the prior expectations that the organization had when we reached out to them. They expected to answer questions and get advice about their current position and how to improve it.

## Self-perception

The results are in line with the self-perception of the organization but do seem a bit too high. Part of the organization is already at the point claimed by the ESA, while some parts are lagging and would score lower. According to the representative, these scores are too high due to his optimistic attitude. To counter this bias in the results, we should consider interviewing multiple employees in the future.

## Serverless Fit

According to the representative, the ESA provides a substantiated and accurate insight regarding an organisation's fit for Serverless. An organization assessed early within their Serverless adoption will benefit from these insights as they become aware of their current maturity. They can determine if it matches their needs and how much effort they require to become ready. It prevents unexpected issues later in the process. This maturity is not limited to Serverless, but all their IT capabilities. It supports the organization in reflecting on whether they have all the necessary capabilities to become a modern IT organization.

## Adoption advice

According to the representative, the advice given within the ESA will improve Serverless adoption. He perceives the proposed steps as logical and valuable investments for better adoption and more effective use of the technology.

## 5.1.2 Improvements

Looking at the expert's opinion, we can state that the ESA effectively supports and improves Serverless adoption within enterprise organizations. The ESA is well executable, relevant, and delivers valuable results in line with the experience of an advanced organization. We did, however, find several improvements.

## Methodology

The representative answered several questions with a score instead of a definite yes or no during the questionnaire. He stated that he expected a scoring scale for all questions. On top of that, it is challenging to determine a bias because of the current response method. Therefore, we need to improve the way responses are put into the ESA tool. Because of time limitations, we cannot change the scoring system

within the ESA. However, we were able to change the instructions. We updated this section to make the consultant ask for a score when respondents are uncertain about an answer. The comments section within the ESA provides room for keeping track of these scores. We updated the ESA accordingly, as shown in Figure 24.

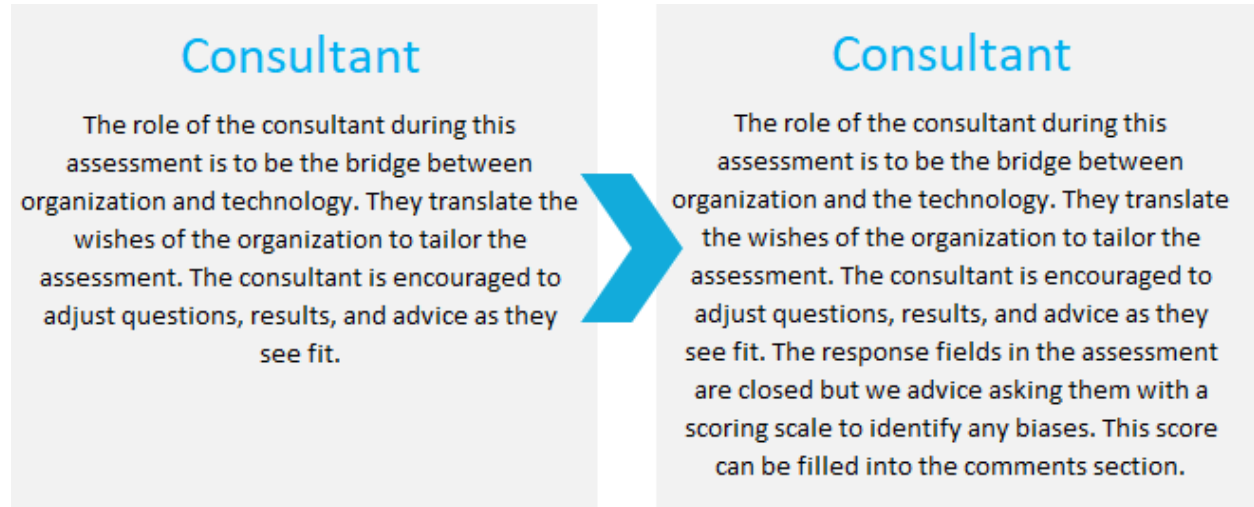


Figure 24: Consultant instructions improvement

## Criteria

The expert mentioned three criteria that he believed were partly incorrect. We made several changes in line with the feedback. The first criterion to discuss is:

*Former: Parties involved with the IT process need to be aware of the advantages and disadvantages of Serverless*

The expert stated that this would not work within their organisation as only some organizational layers are interested in technological matters. Therefore, we adapt the scope of this criterion. The expert explained that only the people directly affected by the technology are interested in these matters, up to and including the IT decision-makers. The consultant must determine what layer this is within the organization they serve:

*New: Parties directly affected by Serverless, up to and including the IT decision-makers, need to be aware of the advantages and disadvantages of Serverless.*

The second criterion that we need to discuss is:

*Former: Extra budget is required for increased governance costs.*

The expert stated that these costs should remain the same if the organization performs their governance correctly. He also did not see an increase during their adoption. When looking within the literature, we cannot find any sources confirming an increase in costs, while some CSPs state that their governance efforts result in cost decreases (Beswick, 2021). Therefore, we felt this criterion was not explicable, and we decided to remove it.

*New: ~~Extra budget is required for increased governance costs~~*

The third criterion that we need to review is:

*Former: The organization must be able to deal with unpredictable infrastructure costs*

The expert stated that the infrastructure costs are only unpredictable during the first year. The organization can use the first year to indicate the succeeding years making cost prediction easier. Therefore, we alter this criterion:

*New: The organization must be able to deal with unpredictable infrastructure costs during the first year of deployment.*

We updated the criteria within the ESA tool in the same manner.

### Self-perception

The results of the demonstration were higher than the self-perception of the organization. Together with the expert, we identified that the interviewee's optimistic attitude caused this difference, resulting in a bias in the results. In the future, the consultants using the ESA must interview multiple employees within the organization to ensure they mitigate this risk. Therefore, we change the instructions of the ESA accordingly, as shown in Figure 25.

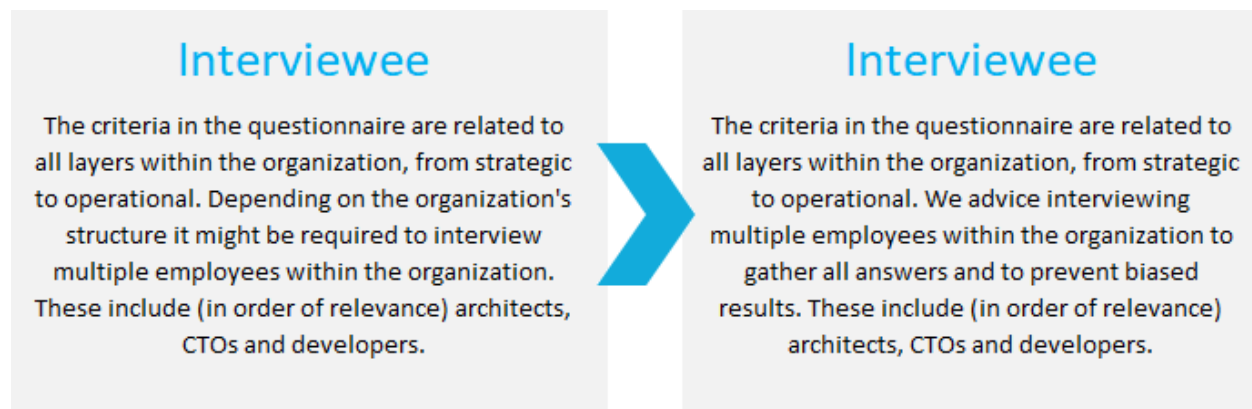


Figure 25: Interviewee instructions improvement

## 5.2 Limitations

This section discusses the limitations to the ESA that we made during the different research methods to deal with the encountered constraints:

### Multi-vocal literature review

To develop a well-founded assessment, we combined theoretical and practical knowledge. We gathered the theoretical knowledge through a multivocal literature research which included grey literature. The grey literature was necessary because the available academic literature fell short when determining the criteria for each assessment topic. While we solved the issue by adding grey literature, it limited the academic substantiation. Grey literature is not peer-reviewed, making it vulnerable to disinformation caused by conflicts of interest from the publishing organizations. To counter this vulnerability, we choose to validate our findings by letting the experts criticize the literature during the first round of the Delphi study.



## Delphi study

For the Delphi study, we gathered a panel of five Serverless experts. Four of these experts were directly or indirectly involved with the commissioning organization. On top of that, one expert is involved with a major CSP. This involvement could lead to a conflict of interest as they would benefit from giving desired answers to further the ESA or make it advise services that the respective CSP offers. We minimized the impact of this conflict by keeping the experts anonymous from one another and letting them validate each other's input. This validation resulted in the removal of many contributions because other experts stated that they were irrelevant.

A second limitation of the Delphi study was the omission of the third round. We chose to skip the third round as the changes were too small to justify an extra one. Because there was no third round, the experts did not validate the final modifications. However, due to the magnitude, we do not believe these changes endanger the credibility of the ESA.

The last limitation of the Delphi study is the determination of the prioritization factors. Because of the limited time available during the Delphi sessions, it was impossible to let the experts judge every criterion individually. Instead, we asked them to choose the ones they believe will have the most impact or require the most effort. By combining their input, we determined what criteria to prioritize. We were, however, unable to discuss each criterion individually, which might have led to flawed priorities. Therefore, we solely add the prioritization score as an indication and make the consultant in charge of choosing the most appropriate actions. Consequently, we believe the used method is satisfactory.

## Case study & Expert opinion

During the case study, we demonstrated the ESA within a single organization. We reached out to three organizations, but two were unable to participate. As Wieringa (2014) described, a single case validation is satisfactory and robust when using it within practice conditions as we did within the organization. These conditions imply that the ESA will have equivalent results in organizations with similar architectures (Wieringa, 2014). In reality, not all architectures are identical, making the validity questionable for different types of organizations. Wieringa (2014) suggests moving to statistical difference-making experiments to validate the research further. This move requires us to perform multiple samples of the validation. These samples enable us to average out the nondeterminism and make it plausible that the ESA works for the entire enterprise population.

To evaluate the ESA, we used the expert opinion method. By interviewing an expert at the demonstration organization, we determined if the ESA's results aligned with their experiences. The limitation of this method is that we could not measure the results. In an ideal situation, we perform two complete adoption processes, one with the ESA's insights and one without. We can then compare both and measure the results. Because this adoption process can take years, this is impossible within the timeframe of this research. According to Wieringa (2014), the expert opinion method is a suitable evaluation method but has limitations: The experts might give positive opinions because of socially desirable remarks (Wieringa, 2014). We encouraged giving negative opinions to counter this limitation as they provide improvement opportunities (Wieringa, 2014).



# 6. Conclusion & Recommendations



## Completing the research and defining what is next

This chapter presents the conclusion of the research in Section 6.1. Hereafter, we give recommendations to Capgemini Invent in Section 6.2 and propose future research in Section 6.3.

### 6.1 Conclusion

This research aimed to deep-dive into the growing market trend of Serverless adoption within enterprises. Investigation indicated that organizations struggle to adopt Serverless due to a lack of guidance, and Capgemini Invent NL had not capitalized on this opportunity yet. We discovered a research gap among existing Serverless frameworks that all kept the initial requirements analysis phase out of scope. Because other sources state this phase to be essential for successful Serverless adoption, we aimed to fill this gap within this study. To align this research with the services of Capgemini Invent, we focused on Enterprise-scale organizations. More specific, when can Serverless enhance an organisation's strategy to deliver business value, and what capabilities does the organisation require for successful adoption. This problem statement led to the following research question:

*How can an enterprise assess its fit with Serverless technology?*

We found that organizations can assess their fit with Serverless technology by determining their strategic fit and organizational readiness using the comprehensive Enterprise Serverless Assessment (ESA) we developed. We realized this assessment by criticizing the knowledge available in the literature and complementing it with the practical knowledge of subject matter experts. This method led to a list of criteria that we structured within a tool that enables consultants to assess organizations effortlessly. By filling in a questionnaire, the tool calculates the required scores and determines if there is a fit between the organization's strategic goals and the benefits of Serverless. If there is a fit, the tool indicates if the organization has the right capabilities or if they need to perform preparatory steps. The tool then highlights areas that require additional attention and proposes an advisory report on how the organization can improve its readiness.

By letting the experts verify each other's input, we ensured that every addition to the list was well-substantiated. They also criticized the theoretical findings in the literature on whether these hold in practice and if they are relevant at an enterprise scale. We then demonstrated the ESA within a multinational financial service provider to validate its usability. Based on the findings during this demonstration, we evaluated the assessment's effectiveness by letting an experienced employee within the organization reflect if the results align with their experiences. This expert stated that the assessed areas were comprehensive and that the criteria were relevant. The results aligned with their self-perception, and the advice was deemed insightful. Therefore, the expert declared that the ESA effectively solves the guidance problem and will support the adoption of Serverless within organizations.

To conclude, our study successfully fills the discovered research gap among Serverless frameworks for enterprise-scale organizations. We substantiated grey and white literature through our expert panel and, in the process, (partly) solved multiple other open research issues, advancing the scientific state of Serverless.

## Scientific contribution

This research contributes to the scientific field of Information Systems in several ways. In the first place, by filling the research gap that we discovered among existing Serverless frameworks for enterprise-scale organizations. While frameworks regarding Serverless do exist, they skip the first of seven phases within the typical application lifecycle. Preliminary research stated that the initial requirements analysis phase was out of scope in all available frameworks. This absence was alarming as other sources claim that determining the fit with Serverless is essential for its success. When doing further exploratory research, we found that most of the existing literature was mainly concerned with technological aspects while the business factors were often understudied. Throughout this research, we filled this gap for enterprise-scale organizations by determining what high-level requirements organizations can meet using Serverless. By exploring and criticizing the motives for using Serverless, we identified the business value that Serverless delivers and when an organization should use it, enabling us to perform the missing requirements analysis phase at an overarching level within organizations. To complete this analysis, we determined what capabilities an organization must have to become ready for using Serverless. To conclude, we are among the first to focus on Serverless's delivered business value and required organizational capabilities compared to preceding studies that were mainly related to the technological aspects.

We made a second contribution by closing the gap between existing white and grey literature. While preliminary studies claimed that grey literature is more advanced, it lacks the academic substantiation that white literature has. We closed this gap by letting experts review the existing theoretical knowledge available in both types of literature. We refuted existing claims through this method, improving the credibility of the confirmed parts and removing those that should not be there. This approach substantiated the grey literature that, through our expert panel, can now be used in future scientific research. On top of that, we complemented the literature with practical knowledge from experts to create a comprehensive perspective. Altogether, we furthered the state of the literature regarding Serverless by criticizing, substantiating, and complementing grey and white literature.

Because of this broad approach, we tackled multiple open research issues in the process. We examined many Serverless aspects that needed further investigation, such as what skills employees require, testing complexity, security risks, and monitoring related matters (Sadaqat, Colomo-Palacios, & Knudsen, 2018). To illustrate the extent of this research, we (partly) answer all the open research problems found by Baldini et al. (2017) using our findings:

- *What are the boundaries of Serverless? Is the scope broader than just FaaS?*  
Our findings confirm the need to answer this question as many interpretations exist in the literature and among experts. We found that it is more expansive than just FaaS and proposed a formal definition for Serverless: A combination of FaaS and BaaS services.
- *Is tooling for Serverless fundamentally different from existing solutions?*  
We found that organisations require additional tools when using Serverless, primarily for monitoring, testing, version control, and CICD.
- *Can legacy code be made to run Serverless?*  
We found that it is possible to migrate existing applications to Serverless, but some experts believe organizations will use Serverless mainly for new-build applications.

- *Is Serverless fundamentally stateless?*  
We verified that the existing services are stateless, but workarounds exist, such as the externalized state pattern.
- *Will there be patterns for building serverless solutions?*  
We found multiple patterns that help and support building Serverless applications for both architects and developers.
- *Does Serverless extend beyond traditional cloud platforms?*  
We discovered that CSPs provide Serverless solutions that run outside the traditionally defined data centres by combining Serverless with IoT and Edge computing.

On top of these open Serverless issues in research, we discovered that our findings also benefit other fields of study. Our assessment results provided insights into the whole IT maturity of an organization. It included a wide range of organizational capabilities and discovered commonalities with other technological innovations such as IoT and Edge Computing.

To sum up, our research filled the found research gap for enterprise-scale organizations and solved multiple open issues within the literature by substantiating grey literature. It enables subsequent studies to investigate the succeeding steps in the Serverless adoption process and integration with other technologies such as IoT and Edge computing due to the commonalities we found. Ultimately, this research paves the way for succeeding studies that will shape the entire Serverless adoption journey within enterprises.

### Practical contribution

This research makes a practical contribution to enterprise organizations and management/IT consultancies. Market research shows that most enterprises aim to adopt Serverless in the upcoming two years but face challenges. Our study provides the guidance they need to adopt Serverless. The developed ESA lets them verify if Serverless delivers the value they need to reach their business goals and if they have the required capabilities for successful adoption. On top of that, they get advice with prioritized steps to improve their Serverless readiness. The ESA structures all motives and the required capabilities within a tool. This tool allows organisations to gather the required information and determine their strategic fit and organizational readiness scores. It can directly advise the organization whether Serverless is the right innovation for them, providing them with the first portion of guidance they need.

Consulting firms can use the ESA to serve their customers in their attempts to become cloud-native and adopt Serverless. They can use the findings to show thought leadership towards their clients and strengthen their partnerships. Because their clients will also need guidance during the rest of the Serverless adoption journey, new business opportunities for these consultancy firms emerge. These firms can support their clients in acquiring the required capabilities or during succeeding steps within the adoption process. They can even extend these opportunities towards their IoT and Edge Computing offerings as their clients can enable these innovations through their Serverless adoption.

To conclude, ESA delivers the insights that organizations need to kickstart their adoption and capitalize on the benefits of Serverless. This kickstart leads to new opportunities for consultancies to serve their clients.

## 6.2 Recommendations

Enterprises plan to start using Serverless technology but face challenges. These customers' main challenge is finding the necessary guidance during the process. Invent can capitalize on this opportunity, leading to the following recommendations:

- Extend current cloud offerings with the ESA. This extension makes customers aware that Invent is a thought leader for innovative cloud technologies. This awareness strengthens the ongoing partnerships with customers and opens the door to new opportunities.
- Develop a full-fledged Serverless offering with the ESA as the entry point. After assessing a customer, the opportunity arises for Invent to become their partner throughout the rest of the adoption. To do this properly, Invent must develop resources for their consultants. A first step could be a decision tree that determines what applications within the Enterprise's landscape to migrate to Serverless.
- Search for commonalities with other offerings and business units within Capgemini Invent. For example, Serverless works better with agile teams, and Capgemini Invent offers agile transformations. On top of that, Serverless enables IoT. Ensure that the IoT unit integrates the new Serverless offering within their proposals.
- Start assessing more organizations. By performing these assessments, you realize an industry-standard score. Customers are eager to know how they compare to these standards to determine if they are ahead or behind the competition. Consultants can directly capitalize on the results by showing the benefits of Serverless and propose the rest of Invent's Serverless offering.

## 6.3 Future Research

We concluded that this research paves the way for succeeding studies that will shape the entire Serverless adoption journey within enterprises. Enterprises still need guidance throughout the rest of this journey, for which we require future research:

- Perform studies on all succeeding steps in the Serverless adoption journey to get an all-embracing method for adopting Serverless. We expect that the following steps are necessary:
  - To further prepare for the transformation phase: Realize a decision tree to systematically determine what applications or parts of applications the organization should migrate to Serverless. This tree must work automatically and with easily accessible data because of the many applications within an enterprise.
  - To start the transformation phase: Realize a roadmap for a structured migration/realization of Serverless applications, focusing on large-scale endeavours.
  - To mature the transformation phase: Realize a Serverless maturity framework for advancing adoption within organizations.
- Perform research on how organizations can obtain the required readiness capabilities we found. Our research primarily determined the necessities for enterprise Serverless adoption, but our focus was less on how organizations can realize them. Some examples could be:
  - Study how organizations can deal with Serverless's fluctuating infrastructure costs and how they can predict them after the first year.

- Study how organizations can stop any manual Cloud deployment configurations and fully transfer towards infrastructure as code.
- Study what policies the IT landscape needs for adding new Serverless components.

Apart from further extending the research, we propose some improvements to our study. While we developed and validated the ESA during the study to solve the research problem, we had to deal with the time constraints of a Master's thesis. These constraints resulted in some limitations that further research can solve:

- During the second round of the Delphi study, we gathered input from the experts to determine the prioritization scores. Because there was no time to assess the impact and effort of each criterion, these scores are less substantiated. During future research, better substantiating these scores could improve the trustworthiness of the advice given within the ESA tool.
- While we determined all relevant capabilities that support Serverless adoption, we could not decide on the minimum requirements. We tried to derive this from the input of the experts, but they were not able to provide us with this information. Therefore, the organisational readiness score does not have a minimum value. The consultant must determine if they believe it is sufficient to start the adoption. When future research shows what capabilities an organization must at least have, we can use these within the tool to determine a more substantiated readiness score.
- We validated the ESA within a single organization. But, as described in Section 5.2, not all organizations are the same. These differences might cause the ESA to work within one organization but fail within another. Moving towards a statistical difference-making experiment in the future ensures we average out any nondeterminism.
- Because Serverless adoption takes years, it was impossible to measure the long-term impact of the ESA. We used an expert's opinion to evaluate the results to solve this problem. However, the expert might have been giving desirable results. To improve the evaluation, we can examine two adoption processes in the future. By applying the ESA within one, we can see if it improves the process compared to the other.





# Bibliography



- Adzic, G., & Chatley, R. (2017). Serverless Computing: Economic and architectural impact. *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering, ESEC/FSE 2017*, 884–889.
- Ahmad, N., Naveed, Q. N., & Hoda, N. (2018). Strategy and procedures for Migration to the Cloud Computing. *2018 IEEE 5th International Conference on Engineering Technologies & Applied Sciences*.
- Albuquerque Jr., L. F., Silva Ferraz, F., Oliveira, R. F., & Galdino, S. M. (2017). Function-as-a-Service X Platform-as-a-Service: Towards a Comparative Study on FaaS and PaaS. *ICSEA*.
- Amazon Web Services. (2020). *Coca-Cola Freestyle Launches Touchless Fountain Experience in 100 Days Using AWS Lambda*. Retrieved 12 5, 2021, from Amazon Web Services: <https://aws.amazon.com/solutions/case-studies/coca-cola-freestyle/>
- Amazon Web Services. (2021). *Amazon API Gateway*. Retrieved November 23, 2021, from Amazon Web Services: <https://aws.amazon.com/api-gateway/>
- Amazon Web Services. (2021). *Amazon Cognito*. Retrieved November 29, 2021, from Amazon Web Services: <https://aws.amazon.com/cognito/>
- Amazon Web Services. (2021). *AWS Lambda*. Retrieved November 29, 2021, from Amazon Web Services: <https://aws.amazon.com/lambda/>
- Amazon Web Services. (2021). *AWS Pricing Calculator*. Retrieved November 29, 2021, from Amazon Web Services: <https://calculator.aws/#/createCalculator/Lambda>
- Barrett, D., & Heale, R. (2020). What are Delphi studies? *Research made simple*, 68-69.
- Beswick, J. (2021, October 6). *Operating serverless at scale: Implementing governance – Part 1*. Retrieved April 11, 2022, from Amazon Web Services: <https://aws.amazon.com/blogs/compute/operating-serverless-at-scale-implementing-governance-part-1/>
- Blamire, J. (2019, July 23). *How Serverless Is Impacting the IT Landscape*. Retrieved December 10, 2021, from DevOps.com: <https://devops.com/how-serverless-is-impacting-the-it-landscape/>
- Bortolini, D., & Obelheiro, R. R. (2020). Investigating Performance and Cost in Function-as-a-Service Platforms. *Springer Nature Switzerland AG*.
- Capgemini. (2021). *A powerhouse of innovation, design, and transformation*. Retrieved November 15, 2021, from Capgemini: <https://www.capgemini.com/service/invent/>
- Capgemini Invent. (2022). *Business Technology*. Retrieved November 15, 2021, from Capgemini: <https://www.capgemini.com/nl-nl/diensten/invent/enterprise-transformation/business-technology/>
- Cloudflare. (2022). *What is serverless computing? | Serverless definition*. Retrieved January 20, 2022, from Cloudflare.com: <https://www.cloudflare.com/learning/serverless/what-is-serverless/>

- Cui, Y. (2019, January 11). *You are thinking about serverless costs all wrong*. Retrieved December 10, 2021, from TheBurningMonk: <https://theburningmonk.com/2019/01/you-are-thinking-about-serverless-costs-all-wrong/>
- Digital Science & Research Solutions, Inc. (2021, October 27). *"Serverless Computing" in Publications*. Retrieved from Dimensions: [https://app.dimensions.ai/discover/publication?search\\_mode=content&search\\_text=%22Serverless%20Computing%22&search\\_type=kws&search\\_field=full\\_search](https://app.dimensions.ai/discover/publication?search_mode=content&search_text=%22Serverless%20Computing%22&search_type=kws&search_field=full_search)
- Eivy, A., & Weinman, J. (2017). Be Wary of the Economics of "Serverless" Cloud Computing. *IEEE Cloud Computing*, 6-12.
- Gannon, D., Barga, R., & Sundaresan, N. (2017). Cloud-Native Applications. *IEEE Cloud Computing*, 16-21.
- Goncharov, I. (2017, November 30). *Serverless Architectures Comparison, Pros & Cons, and Case Studies*. Retrieved December 10, 2021, from Agile Engine: <https://agileengine.com/serverless-architecture/>
- Google. (2021). *Firebase*. Retrieved November 28, 2021, from Google Developers: <https://firebase.google.com/>
- Graw, M. (2021, March 23). *On-prem vs cloud storage: Can cloud ever be as secure?* Retrieved December 15, 2021, from Tom's Guide: <https://www.tomsguide.com/features/on-prem-vs-cloud-storage-can-cloud-ever-be-as-secure>
- Heerkens, H. W. (2017). Solving Managerial Problems Systematically.
- Hellerstein, J. M., Faleiro, J., Gonzalez, J. E., Schleier-Smith, J., Sreekanti, V., Tumanov, A., & Wu, C. (2019). Serverless Computing: One Step Forward, Two Steps Back. *CIDR'19*.
- Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design Science in Information Systems Research. *MIS Quarterly Vol. 28 No. 1*, 75-105.
- IBM Market Development & Insights. (2021). Serverless in the enterprise, 2021: Building the next generation of efficient, flexible, cost-effective. *IBM Market Development & Insights*.
- ISACA. (2018). *COBIT 2019 Framework: Governance and Management Objectives*.
- Jonas, E., Schleier-Smith, J., Sreekanti, V., Khandelwal, A., Pu, Q., Carreira, J., . . . Patterson, D. A. (2019). Cloud Programming Simplified: A Berkeley View on Serverless Computing. *UC Berkeley*.
- Kächele, S., Spann, C., Hauck, F. J., & Domaschka, J. (2013). Beyond IaaS and PaaS: An Extended Cloud Taxonomy for Computation, Storage and Networking. *IEEE/ACM 6th International Conference on Utility and Cloud Computing*.
- Kratzke, N., & Quint, P.-C. (2017). Understanding cloud-native applications after 10 years of cloud computing - A systematic mapping study. *The Journal of Systems and Software* 126, 1-16.
- Kritikos, K., & Skrzypek, P. (2018). A Review of Serverless Frameworks. *IEEE/ACM International Conference on Utility and Cloud Computing Companion (UCC Companion)*.

- Kumar, V. (2020, October 6). *Does my cloud strategy have a silver lining?* Retrieved from Capgemini.com: <https://www.capgemini.com/dk-en/2020/10/does-my-cloud-strategy-have-a-silver-lining/>
- Lefèvre, X. (2020, July 30). *Is serverless cheaper for your use case? Find out with this calculator.* Retrieved December 15, 2021, from Medium: <https://medium.com/serverless-transformation/is-serverless-cheaper-for-your-use-case-find-out-with-this-calculator-2f8a52fc6a68#de0f>
- Leitner, P., Wittern, E., Spillner, J., & Hummer, W. (2019). A mixed-method empirical study of Function-as-a-Service software development in industrial practice. *The Journal of Systems and Software* 149 (2019), 340-359.
- Logicata. (2021). *What are the 6 R's of Cloud Migration?* Retrieved December 5, 2021, from Logicata: <https://www.logicata.com/blog/what-are-the-6-rs-of-cloud-migration/>
- Manner, J., Endreß, M., Heckel, T., & Wirtz, G. (2018). Cold Start Influencing Factors in Function as a Service. *IEEE/ACM International Conference on Utility and Cloud Computing Companion (UCC Companion)*.
- Maryland.gov. (2022). *Phase 4: Requirements Analysis - Custom Single Release Project.* Retrieved from Marland.gov: <https://doit.maryland.gov/SDLC/Custom/Pages/Phase04Single.aspx#:~:text=The%20purpose%20of%20the%20requirements,%2C%20and%20stakeholder%2Dapproved%20requirements.>
- MindTools. (2022). *The Action Priority Matrix.* Retrieved March 29, 2022, from MindTools: [https://www.mindtools.com/pages/article/newHTE\\_95.htm](https://www.mindtools.com/pages/article/newHTE_95.htm)
- Nupponen, J., & Taibi, D. (2020). Serverless: What it Is, What to Do and What Not to Do. *2020 IEEE International Conference on Software Architecture Companion (ICSA-C)*.
- Peffer, K., Tuunanen, T., Rothenberger, M. A., & Samir, C. (2007). A Design Science Research Methodology for Information Systems Research. *Journal of Management Information Systems* , 45-77.
- Pekkala, A. (2019). Migrating a web application to serverless architecture. *University of Jyväskylä*.
- Rajan, A. P. (2020). A review on serverless architectures - function as a service (FaaS) in cloud computing. *TELKOMNIKA Telecommunication, Computing, Electronics and Control*, 530-537.
- Roberts, M. (2018, May 22). *Serverless Architectures.* Retrieved from martinFowler: <https://martinfowler.com/articles/serverless.html>
- Sadaqat, M., Colomo-Palacios, R., & Knudsen, L. E. (2018). Serverless Computing: A Multivocal Literature Review. *NOKOBIT - Norsk konferanse for organisasjoners bruk av informasjonsteknologi*.
- Sewak, M., & Singh, S. (2018). Winning in the era of Serverless Computing and FaaS. *I2CT*.
- Soldani, J., Yussupov, V., Breitenbücher, U., Brogi, A., & Leymann, F. (2020). FaaS: Your Decisions: Classification Framework and Technology Review of Function-as-a-Service Platforms. *CoRR*.
- Stefanac, T., & Colomo-Palacios, R. (2021). NoOps – A Multivocal literature review. *Elsevier*.

- Tayal, A., Lam, E., Choudhury, D., Dickerson, M., Moovera, G., & Arora, G. (2019). Determining the Total Cost of Ownership of Serverless Technologies when compared to Traditional Cloud. *Deloitte Consulting*.
- Van Dijk, F. W. (2017). *Adopting the Cloud: A multi-method approach towards developing a cloud maturity model*. Enschede: University of Twente.
- Villamizar, M., Garcés, O., Ochoa, L., & Castro, H. (2017). Cost comparison of running web applications in the cloud using monolithic, microservice, and AWS Lambda architectures. *Service Oriented Computing and Applications* 11(6), 233-247.
- Von Laszewski, G., Diaz, J., Wang, F., & Fox, G. (2012). Comparison of Multiple Cloud Frameworks. *2012 IEEE Fifth International Conference on Cloud Computing*, 734-741.
- Wagner, B., & Sood, A. (2016). Economics of Resilient Cloud Services. *1st IEEE International Workshop on Cyber Resilience Economics*.
- Wieringa, R. J. (2014). *Design Science Methodology*. Enschede: Springer.
- Wolf, O. (2021). *Serverless Architecture in Short*. Retrieved December 15, 2021, from Specify.io: <https://specify.io/concepts/serverless-baas-faas>

# Appendix 1: Delphi Details



This appendix contains details regarding the contribution of the experts during the different Delphi rounds. The tables present the number of experts per criterion: favour, against, or requiring changes.

## Round 1

Criterion	Favour	Against	Change	Update
<b>Target Picture</b>				
Does the organization aim for, or require, improved elasticity and agility for their applications?	4	1	0	None
Does the organization aim for, or require, a lower total cost of ownership for their applications?	3	1	1	Modified
Does the organization aim for, or require, a faster time to market for their applications?	4	1	0	None
Does the organization aim for, or require, reduced energy consumption for their applications?	2	3	0	Removed
Does the organization aim for, or require, improved security for their applications?	1	4	0	Removed
<b>Architecture Framework</b>				
Does the organization have or plan applications with service-oriented architectures?	4	0	1	Modified
Does the organization emphasize reusing functionality?	3	1	1	Modified
Does the organization have applications with predictable execution times?	1	2	2	Modified
<b>Target Operating Model and Governance</b>				
Are business and IT aligned within the organization?	1	3	0	Removed
Does the organization deploy a Cloud operating model?	5	0	0	None
Does the organization use a DevOps methodology?	5	0	0	None
Does the organization use a DevSecOps methodology?	5	0	0	None
Does the organization use a FinOps methodology?	5	0	0	None
Is capacity available for increased technology governance?	1	4	0	Removed
Is capacity available for function governance?	3	0	2	Modified
<b>Business Change Management</b>				
Are relevant stakeholders aware of the Serverless advantages?	2	0	3	Modified
Are developers experienced with Functional Programming?	1	3	1	Removed
Are developers experienced with the Immutable Infrastructure Paradigm?	2	2	1	Modified
Are developers experienced with the Common Serverless Patterns?	2	2	1	Modified
Are developers experienced with Interpreted Programming Languages?	1	3	1	Removed
Do developers feel responsible for the financial performance of their applications?	2	1	2	Modified
<b>Security and Compliance</b>				
Are developers able to validate and encrypt all communications?	4		1	Modified
<b>Financial Implications Analysis</b>				
Are financial buffers available for unintended high infrastructure costs?	4	0	1	Modified
<b>Cloud Deployment</b>				
Can the organization accept a more severe vendor lock-in?	4	0	1	Moved

Does the organization have partnerships with a CSP?	1	3	0	Removed
Does the organization use database, API gateway, logging, and IaaS services provided by a CSP?	3	0	1	Modified
Does the organization develop with programming languages that a CSP supports?	2	0	1	Removed
Does the organization have testing and CI/CD tools in place?	2	0	2	Modified

Table 7: Delphi Details - Round 1

## Round 2

Criterion	Favor	Against	Change	Update
<b>Target Picture</b>				
Does the organization aim for, or require, a faster time to market for their applications?	5	0	0	None
Does the organization aim for, or require, improved elasticity and agility?	5	0	0	Modified
Does the organization aim for, or require, a lower TCO for their applications in the long run?	4	0	2	None
Does the organization aim for, or require, increased application availability?	5	0	0	None
Does the organization aim for, or require, enabling future IT innovations? I.e. IoT or Edge	5	0	0	None
Does the organization aim for, or require, more efficient use of (human) resources?	3	0	2	Modified
<b>Architecture Framework</b>				
Does the organization have/plan applications with modular/distributed architectures?	5	0	0	None
Does the organization facilitate the re-use of functionality/code/services?	5	0	0	None
Is there a streamlined definition within the organization regarding the scope of Serverless?	2	0	2	Modified
Do (solution) architects use their CSP's well-architected framework?	5	0	0	None
Do (solution) architects know the Serverless common/best practices	2	0	2	Modified
Do (solution) architects have experience with event-driven architectures?	5	0	0	None
Do (solution) architects have insights into the performance of the applications? For example, through performance tests?	5	0	0	None
Do (solution) architects keep function points low?	4	0	1	Modified
<b>Target Operating Model and Governance</b>				
Does the organization deploy a Cloud Operating Model?	5	0	0	None
Does the organization have distributed structure?	3	0	2	Modified
Does the organization use a DevOps methodology?	5	0	0	None
Does the organization use a DevSecOps methodology?	5	0	0	None
Does the organization use a FinOps methodology?	5	0	0	None
Does the organization have delivery teams with end-to-end responsibility?	2	0	3	Modified
Does the organization encourage a platform-based approach? I.e. using templates to realize new solutions quickly and know where to position them.	5	0	0	None
Does the organization prioritize the usage of standards supported by multiple Cloud Providers?	5	0	0	None
Is capacity available for increased information governance? I.e. what team manages what information?	5	0	0	None
<b>Business Change Management</b>				

Are the organizational layers involved with the IT process familiar with the advantages and disadvantages of Serverless? (Compared to other Cloud solutions such as IaaS or PaaS)	5	0	0	None
Is Serverless included in the Cloud/Platform/Architecture strategies? I.e. are architects/analysts/developers aware that Serverless is an option?	5	0	0	None
Are business teams familiar with IT responsibility?	5	0	0	None
Do development teams work in an Agile way?	5	0	0	None
Do development teams have insights into the financial performance of their applications?	5	0	0	None
Are development teams familiar with the Common Serverless Patterns?	5	0	0	None
Are development teams familiar with the Immutable Infrastructure Paradigm?	5	0	0	None
Are development teams familiar with the Cloud-Native mindset?	5	0	0	None
Are development teams familiar with Separation of Concerns?	5	0	0	None
Does the organization realize innovative projects that are close to business?	2	3	0	Removed
Do development teams feel a purpose towards the product and organization?	3	2	0	Removed
<b>Security and Compliance</b>				
Does the organization have an expertise department where security advisory is available? Scale can depend on the organizational structure of the organization.	5	0	0	None
Are the organization's development teams capable of securing communications and validating in/outputs?	5	0	0	None
Are development teams aware of the risks caused by public interfaces?	5	0	0	None
Are policies in place for adding new Serverless components to the IT landscape?	5	0	0	None
Are code based compliance and security checks imposed within the Ci/Cd pipeline?	5	0	0	None
Can the organization deal with difficult logging and auditing processes?	3	0	2	Modified
Do the development teams get extra security by design training?	2	2	0	Removed
Are all components within the organization protected with networks?	1	4	0	Removed
<b>Financial Implications Analysis</b>				
Can the organization make a high initial investment?	1	0	4	Modified
Can the organization invest when costs are difficult to assess? I.e. unpredictable cost fluctuations?	5	0	0	None
Does the organization's financial model allow an invocation based billing model?	5	0	0	None
Is the budget available for extra training and recruitment?	5	0	0	None
Is the budget available for extra governance?	5	0	0	None
Is the budget available for extra security?	2	1	2	Removed
Are quotas available for traffic costs during development by inexperienced development teams?	2	0	2	Modified
<b>Cloud Deployment</b>				
Does the organization already use the database, API gateway, logging, and monitoring services the CSP provides?	3	0	2	Modified
Does the organization use testing, Ci/Cd, and version control tools?	5	0	0	None
Does the organization enforce Infrastructure as Code and Immutable Infrastructure?	5	0	0	None
Does the organization have a central logging system in place?	2	1	2	Modified

Table 8: Delphi Details - Round 2





# Appendix 2: Delphi Consensus



This appendix contains the final Delphi consensus translated to the criteria used in the assessment.

## Target Picture

- Does the organization aim for, or require, a faster time to market for their applications?
- Does the organization aim for, or require, improved elasticity and agility?
- Does the organization aim for, or require, a lower TCO for their applications?
- Does the organization aim for, or require, increased application availability?
- Does the organization aim for, or require, enabling future IT innovations? I.e. IoT or Edge
- Does the organization aim for, or require, more efficient use of (human) resources?

## Architecture Framework

- Does the organization have/plan applications with modular/distributed architectures?
- Does the organization facilitate the re-use of functionality/code/services?
- Is there alignment within the organization regarding the scope of Serverless? I.e. what service offerings are included, and how are they used?
- Do (solution) architects use their CSP's well-architected framework?
- Do (solution) architects know the Serverless common/best practices and patterns
- Do (solution) architects have experience with event-driven architectures?
- Do (solution) architects have insights into the performance of the applications? For example, through performance tests?
- Do (solution) architects keep application complexity low? I.e. Microservices over significant accumulations of functionality?

## Target Operating Model and Governance

- Does the organization deploy a Cloud Operating Model?
- Does the organization operate without organizational silos? I.e. Agile at scale
- Does the organization use a DevOps methodology?
- Does the organization use a DevSecOps methodology?
- Does the organization use a FinOps methodology?
- Does the organization have delivery teams with end-to-end responsibility? I.e. BizDevSecOps, Fusion Teams, or Product Oriented Delivery
- Does the organization encourage a platform-based approach? I.e. using templates to realize new solutions quickly and know where to position them.
- Does the organization prioritize the usage of standards supported by multiple Cloud Providers?
- Is capacity available for increased information governance? I.e. what team manages what information?

## Business Change Management

- Are the organizational layers involved with the IT process familiar with the advantages and disadvantages of Serverless? (Compared to other Cloud solutions such as IaaS or PaaS)

- Is Serverless included in the Cloud/Platform/Architecture strategies? I.e. are architects/analysts/developers aware that Serverless is an option?
- Are business teams familiar with IT responsibility?
- Do development teams work in an Agile way?
- Do development teams have insights into the financial performance of their applications?
- Are development teams familiar with the Common Serverless Patterns?
- Are development teams familiar with the Immutable Infrastructure Paradigm?
- Are development teams familiar with the Cloud-Native mindset?
- Are development teams familiar with Separation of Concerns?

### **Security and Compliance**

- Does the organization have an expertise department where security advisory is available? Scale can depend on the organizational structure of the organization.
- Are the organization's development teams capable of securing communications and validating in/outputs?
- Are development teams aware of the risks caused by public interfaces?
- Are policies in place for adding new Serverless components to the IT landscape?
- Are code based compliance and security checks imposed within the Ci/Cd pipeline?
- Are automatic compliance and security validation checks imposed on the Cloud landscape?

### **Financial Implications Analysis**

- Is the organization aware that Serverless is a long term investment, and is it capable of making this investment? I.e. costs will only drop after a long period.
- Can the organization invest when costs are difficult to assess? I.e. unpredictable cost fluctuations?
- Does the organization's financial model allow an invocation based billing model?
- Is the budget available for extra training and recruitment?
- Is the budget available for extra governance?
- Are quotas available for traffic costs during development by inexperienced development teams?

### **Cloud Deployment**

- Does the organization already use the database, API gateway, logging, monitoring, and security services the CSP provides?
- Does the organization use testing, Ci/Cd, and version control tools?
- Does the organization enforce Infrastructure as Code and Immutable Infrastructure?
- Does the organization have a distributed monitoring system in place?

# Appendix 3: Prioritization Details



This appendix contains the details that lead to the prioritization scores. We based these scores on the input given by the experts during the second Delphi round. The numbers indicate how many experts pinpointed that criterion to have a high impact or require effort. These scores translate to their impact/effort ratings, as shown in Table 9. Table 10 shows the number of pinpoints for each of the criteria.

Impact/effort rating	Number of pinpoints
Low	0
Medium	1
High	2-5

Table 9: Impact/effort rating translation

Criterion	Impact	Effort
<b>Architecture Framework</b>		
Does the organization have/plan applications with modular/distributed architectures?	1	2
Does the organization facilitate the re-use of functionality/code/services?	1	0
Is there alignment within the organization regarding the scope of Serverless? I.e. what service offerings are included, and how are they used?	1	0
Do (solution) architects use their CSP's well-architected framework?	2	0
Do (solution) architects know the Serverless common/best practices and patterns	1	0
Do (solution) architects have experience with event-driven architectures?	1	1
Do (solution) architects have insights into the performance of the applications? For example, through performance tests?	0	0
Do (solution) architects keep application complexity low? I.e. Microservices over significant accumulations of functionality?	0	1
<b>Target Operating Model and Governance</b>		
Does the organization deploy a Cloud Operating Model?	2	1
Does the organization operate without organizational silos? I.e. Agile at scale	2	2
Does the organization use a DevOps methodology?	3	2
Does the organization use a DevSecOps methodology?	1	2
Does the organization use a FinOps methodology?	1	2
Does the organization have delivery teams with end-to-end responsibility? I.e. BizDevSecOps, Fusion Teams, or Product Oriented Delivery	2	2
Does the organization encourage a platform-based approach? I.e. using templates to realize new solutions quickly and know where to position them.	2	1
Does the organization prioritize the usage of standards supported by multiple Cloud Providers?	1	0
Is capacity available for increased information governance? I.e. what team manages what information?	0	0
<b>Business Change Management</b>		
Are the organizational layers involved with the IT process familiar with the advantages and disadvantages of Serverless? (Compared to other Cloud solutions such as IaaS or PaaS)	2	1
Is Serverless included in the Cloud/Platform/Architecture strategies? I.e. are architects/analysts/developers aware that Serverless is an option?	3	0
Are business teams familiar with IT responsibility?	1	1
Do development teams work in an Agile way?	1	1
Do development teams have insights into the financial performance of their applications?	0	0
Are development teams familiar with the Common Serverless Patterns?	2	2

Are development teams familiar with the Immutable Infrastructure Paradigm?	1	1
Are development teams familiar with the Cloud-Native mindset?	1	2
Are development teams familiar with Separation of Concerns?	0	1
<b>Security and Compliance</b>		
Does the organization have an expertise department where security advisory is available? Scale can depend on the organizational structure of the organization.	2	0
Are the organization's development teams capable of securing communications and validating in/outputs?	1	0
Are development teams aware of the risks caused by public interfaces?	1	1
Are policies in place for adding new Serverless components to the IT landscape?	0	0
Are code based compliance and security checks imposed within the Ci/Cd pipeline?	1	1
Are automatic compliance and security validation checks imposed on the Cloud landscape?	1	1
<b>Financial Implications Analysis</b>		
Is the organization aware that Serverless is a long term investment, and is it capable of making this investment? I.e. costs will only drop after a long period.	1	0
Can the organization invest when costs are difficult to asses? I.e. unpredictable cost fluctuations?	0	0
Does the organization's financial model allow an invocation based billing model?	0	0
Is the budget available for extra training and recruitment?	0	0
Is the budget available for extra governance?	0	0
Are quotas available for traffic costs during development by inexperienced development teams?	0	0
<b>Cloud Deployment</b>		
Does the organization already use the database, API gateway, logging, monitoring, and security services the CSP provides?	0	1
Does the organization use testing, Ci/Cd, and version control tools?	0	0
Does the organization enforce Infrastructure as Code and Immutable Infrastructure?	0	0
Does the organization have a distributed monitoring system in place?	0	0

*Table 10: Prioritization details*

# Appendix 4: ESA Tool



This appendix contains an overview of the different pages of the assessment tool. The pages are shown in the same order as presented within the tool.



Figure 26: ESA tool - Frontpage

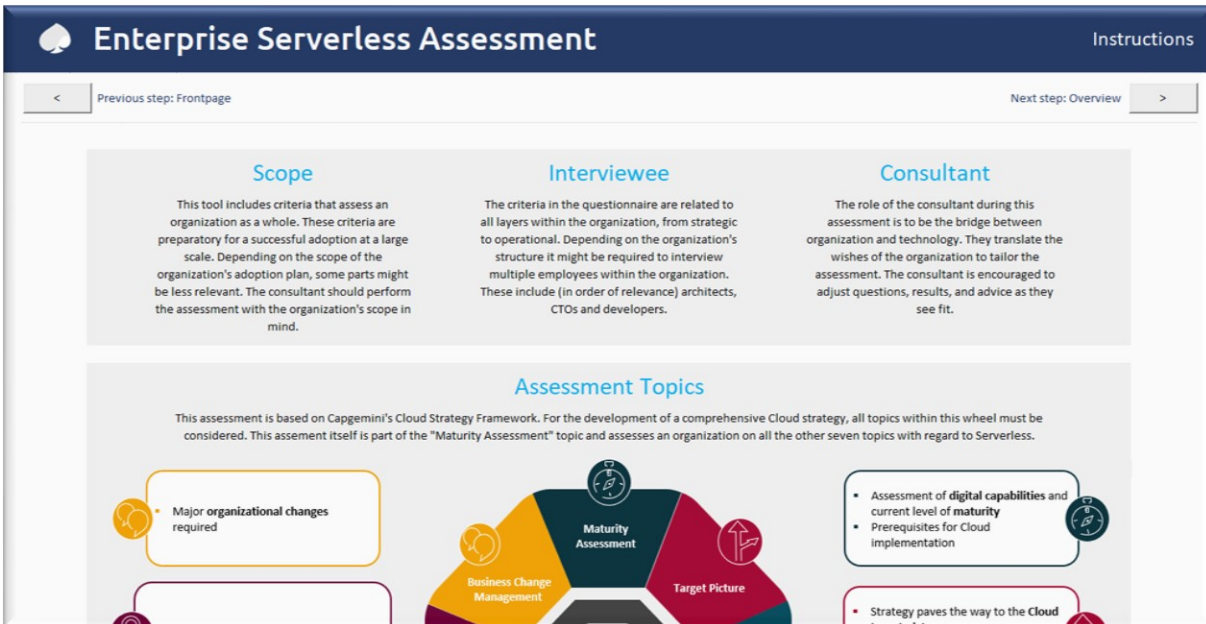


Figure 27: ESA tool - Instructions page

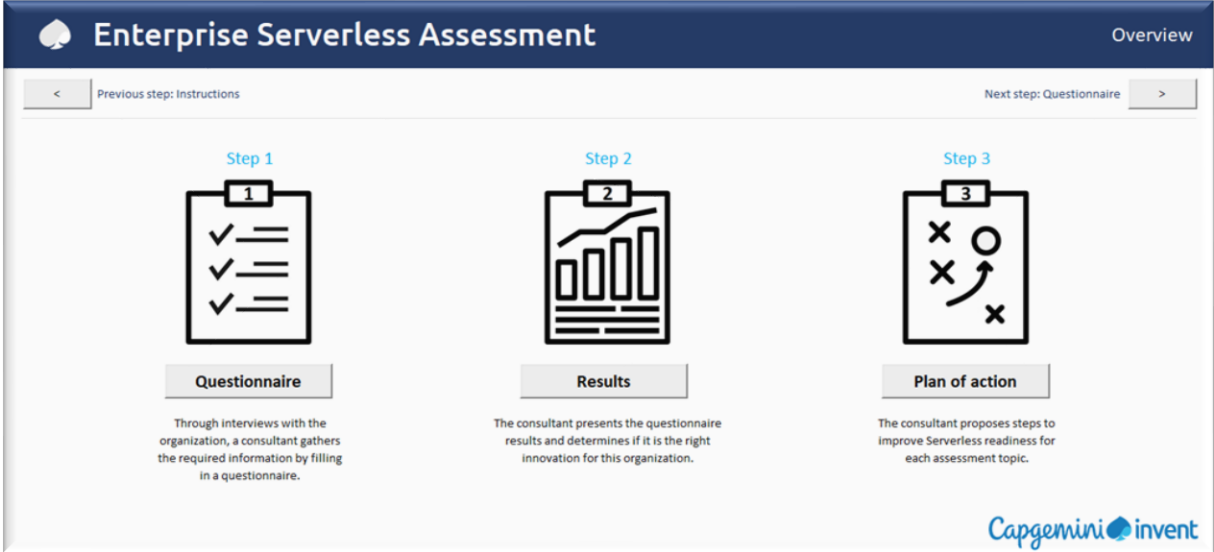


Figure 28: ESA tool - Overview page

**Enterprise Serverless Assessment** Questionnaire

Previous step: Overview | Next step: Results

**Target Picture**  
*(Select criteria cell for additional information)*

*Guidelines for quick decision making*  
*(Comments cells are available to fill in additional information about the answer)*

Criteria	Comments	Response	Impact
Does the organization aim for, or require, a faster time to market for their applications?		FALSE	N/a
Does the organization aim for, or require, improved elasticity and agility?		FALSE	N/a
- Do one or more applications have unpredictable workloads?		-	-
- Do one or more applications have significant idle time?		-	-
Does the organization aim for, or require, a lower TCO for their applications?		FALSE	N/a
- Do development teams spend, on average, more than 25% of their time on maintenance tasks?		-	-
- Does pre-planning applications during development require a significant part of the available capacity?		-	-
- Do applications have a cost-ineffective infrastructure? I.e. a lot of idle time or significant fluctuations in workloads?		-	-
Does the organization aim for, or require, increased application availability?		FALSE	N/a
Does the organization aim for, or require, enabling future IT innovations? I.e. IoT or Edge		FALSE	N/a
Does the organization aim for, or requires, more efficient use of (human) resources?		FALSE	N/a

**Architecture Framework**  
*Requirements for architecture management*

Criteria	Comments	Response	Impact
Does the organization have/plan applications with modular/distributed architectures?		FALSE	High
Does the organization facilitate re-use of functionality/code/services?		FALSE	High

Figure 29: ESA tool - Questionnaire page

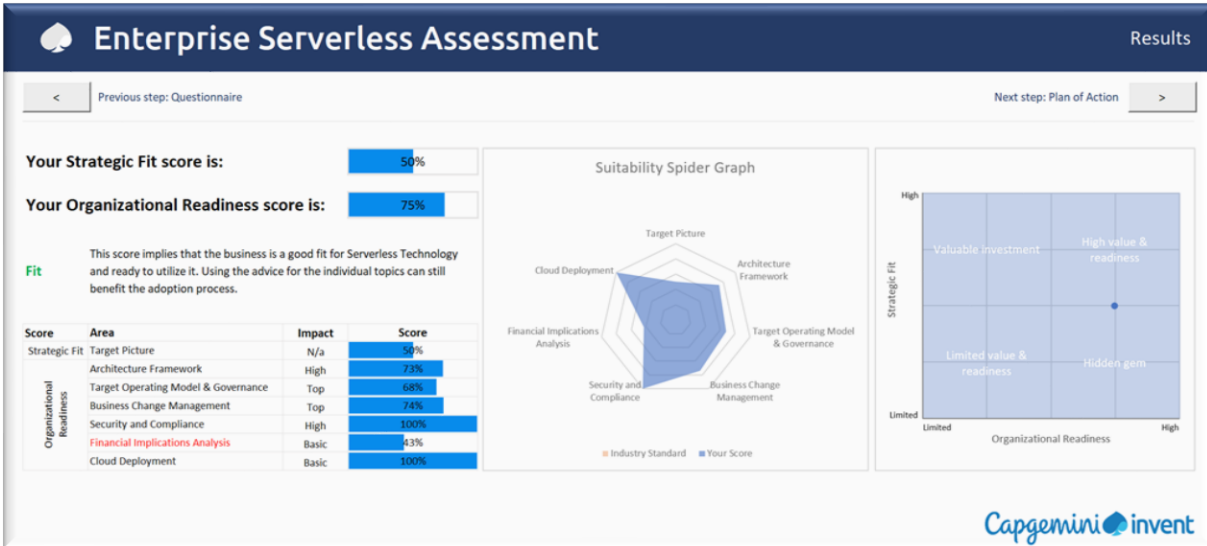


Figure 30: ESA tool - Results page

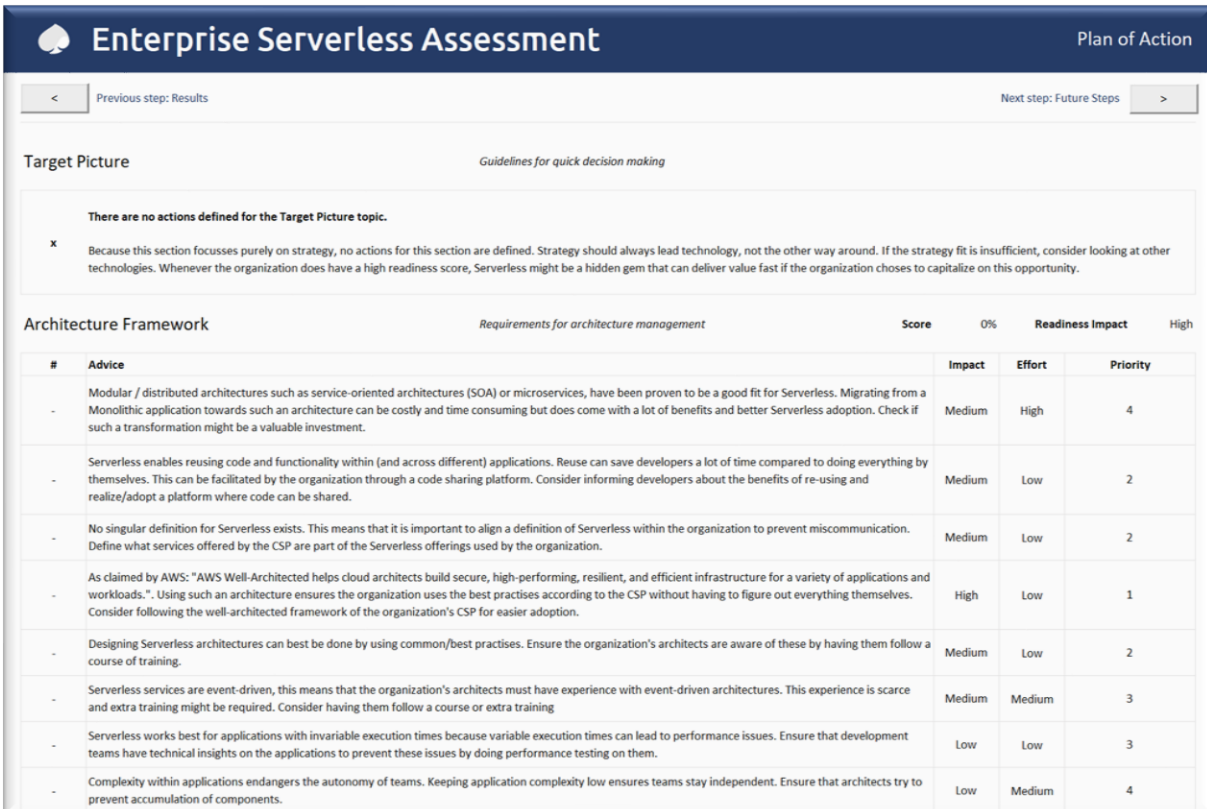


Figure 31: ESA tool - Plan of Action page

**Enterprise Serverless Assessment** Future steps

Previous step: Plan of Action Next step: Epilogue

Now



**Start**

A small application is transformed into Serverless as a wedge offer to show the effect and benefits for the organization. The desired business value is measured to quantify the improvements.

Next



**Structure**

The organization has seen the benefits of Serverless for a single application. It is time to structure the adoption within the organization. These applications are chosen based on importance and compatibility.

Later



**Scale**

The organization has structured their Serverless plans and is ready to use Serverless throughout the organization. The usage is gradually scaled-up and applied in more applications.

**Capgemini invent**

Figure 32: ESA tool - Next steps page


**Enterprise Serverless Assessment** Epilogue

Previous step: Future Steps

**Capgemini Invent's Enterprise Serverless Assessment**

You have reached the end of this assessment tool to determine an organization's strategic fit and organizational readiness for Serverless. When the result was a good or potential fit, and the organization followed the advice in the plan of action, the organization is ready to start using Serverless. This result implies that Serverless will support the organization in achieving business goals, and the organization has mitigated potential challenges. The organization can be confident that rebuilding applications with Serverless technologies will be smooth and deliver the promised benefits.

When the result was a bad fit, other investments are likely to be more worthwhile. The proposed changes can still be used as guidelines to improve Serverless readiness for future innovations.



**Enterprise Serverless Assessment**

**Developed by**  
 Thom Leemans  
[thom.leemans@capgemini.com](mailto:thom.leemans@capgemini.com)

**In cooperation with**  
 Capgemini Invent  
 Daniel Visser  
 Christiaan Tick

University of Twente  
 Dr. Lucas O. Meertens  
 Dr. Faiza A. Bukhsh

**Capgemini invent**

Figure 33: ESA tool - Epilogue page



**Enterprise Serverless Assessment** Settings

[< Previous: Frontpage](#)

### Minimal Score Parameters

General	Min. Strat	Min. Read	Conclusion
Fit	0,5	0,5	This score implies that the business is a good fit for Serverless Technology and ready to utilize it. Using the advice for the individual topics can still benefit the adoption process.
Potential	0,5	0	This score implies that Serverless technology has the potential to support this companies strategy, but it is advised to perform some actions for better adoption. The next section discusses these changes for each of the different topics.
Unfit	0	0	This score implies that Serverless is unlikely to be a suitable investment for this company as the benefits of Serverless do not support the business goals.

### Industry Scores

Area	Score	Manage scores
Cloud Target Picture	0	
Cloud Architecture Framework	0	
Target Operating Model & Governance	0	
Business Change Management	0	
Security and Compliance	0	
Financial Implications Analysis	0	
Cloud Deployment	0	

Figure 34: ESA tool - Settings page

**Enterprise Serverless Assessment** Industry Scores

[< Previous: Settings](#)

	Organization Weight	Average	1	2	3	4	5	6	7	8	9	10	11	...
Cloud Target Picture	0													
Cloud Architecture Framework	15													
Target Operating Model & Governance	22													
Business Change Management	19													
Security and Compliance	12													
Financial Implications Analysis	7													
Cloud Deployment	4													
Readiness		0												

Figure 35: ESA tool - Industry standard settings page



# Appendix 5: Demonstration Details

This appendix contains the questionnaire details gathered and used during the demonstration phase. We gathered the answers to this questionnaire by interviewing a representative from a major financial organization.

## Target Picture

### *Guidelines for quick decision making*

Criteria	Comments	Response
Does the organization aim for, or require, a faster time to market for their applications?	With regards to Cloud: Yes; We do not expect this benefit with Serverless	FALSE
Does the organization aim for, or require, improved elasticity and agility?	Not a direct motive, but a nice-to-have	FALSE
Does the organization aim for, or require, a lower TCO for their applications?	We expect lower costs in development but do not expect to benefit from it at our scale	FALSE
Does the organization aim for, or require, increased application availability?	Not a direct motive, but a nice-to-have	FALSE
Does the organization aim for, or require, enabling future IT innovations? I.e. IoT or Edge	Not a direct motive, but a nice-to-have	FALSE
Does the organization aim for, or require, more efficient use of (human) resources?	The most crucial motive for us. Developers can focus on the business logic	TRUE

## Architecture Framework

### *Requirements for architecture management*

Criteria	Comments	Response
Does the organization have/plan applications with modular/distributed architectures?	They are working hard to realize these architectures.	TRUE
Does the organization facilitate the re-use of functionality/code/services?	Working hard to become API - driven: an essential part of our agenda	TRUE
Is there alignment within the organization regarding the scope of Serverless? I.e. what service offerings are included, and how are they used?	Not defined, but there is an informal definition that causes little discussion. They believe it is the technologies where you do not run the instance yourself. Part of PaaS/FaaS/BaaS	FALSE
Do (solution) architects use their CSP's well-architected framework?	They just started to do so. This action is getting traction.	TRUE
Do (solution) architects know the Serverless common/best practices?	Working on getting there, but a long way to go. On a scale of 1-5: 2	FALSE
Do (solution) architects have experience with event-driven architectures?	Working on getting there, but a long way to go. On a scale of 1-5: 2	FALSE
Do (solution) architects have insights into the performance of the applications? For example, through performance tests?	Competence centre is present - For critical apps, we know everything	TRUE
Do (solution) architects keep application complexity low? I.e. Microservices over significant accumulations of functionality?	They are starting to get there. On a scale of 1-5: 3	TRUE

## Target Operating Model & Governance

### *Transformation to a multi-speed IT organization*

Criteria	Comments	Response
Does the organization deploy a Cloud Operating Model?	Yes, they do.	TRUE

Does the organization operate without organizational silos? I.e. Agile at scale	They are trying to break them but have not succeeded yet.	FALSE
Does the organization use a DevOps methodology?	Yes, they do.	TRUE
Does the organization use a DevSecOps methodology?	Officially they do, but in practice, it does not always hold.	FALSE
Does the organization use a FinOps methodology?	Yes, every team is responsible for their financials.	TRUE
Does the organization have delivery teams with end-to-end responsibility? I.e. BizDevSecOps, Fusion Teams, or Product Oriented Delivery	Yes, some teams have full end-to-end delivery.	TRUE
Does the organization encourage a platform-based approach? I.e. using templates to realize new solutions quickly and know where to position them.	Not for every part, but they have specific departments that use these templates.	TRUE
Does the organization prioritize the usage of standards supported by multiple Cloud Providers?	No, they do not use the poly cloud. One app is at one provider. We are aware of lock-in, but they counter it by ensuring they can transfer their business logic.	FALSE
Is capacity available for increased information governance? I.e. what team manages what information?	Yes, they are working hard to do this.	TRUE

## Business Change Management

### *Required organizational changes*

Criteria	Comments	Response
Are the organizational layers involved with the IT process familiar with the advantages and disadvantages of Serverless? (Compared to other Cloud solutions such as IaaS or PaaS)	Varies enormously: Some domains do have this knowledge, others don't. True up until a specific part of management. Serverless is just a technology.	FALSE
Is Serverless included in the Cloud/Platform/Architecture strategies? I.e. are architects/analysts/developers aware that Serverless is an option?	Yes, They have Serverless > Containers > Others. We are actively marketing it.	TRUE
Are business teams familiar with IT responsibility?	They let the business focus on the business matters; We want to spare them from technology matters.	FALSE
Do development teams work in an Agile way?	Yes.	TRUE
Do development teams have insights into the financial performance of their applications?	Yes, they have. Everyone can see what they are spending.	TRUE
Are development teams familiar with the Common Serverless Patterns?	Some teams do, mainly the progressive people/hobbyists	TRUE
Are development teams familiar with the Immutable Infrastructure Paradigm?	Some teams do, mainly the progressive people/hobbyists	TRUE
Are development teams familiar with the Cloud-Native mindset?	Some teams do, mainly the progressive people/hobbyists	TRUE
Are development teams familiar with Separation of Concerns?	Some teams do, mainly the progressive people/hobbyists	TRUE

## Security and Compliance

### *Mitigation of additional security risks*

Criteria	Comments	Response
----------	----------	----------

Does the organization have an expertise department where security advisory is available? Scale can depend on the organizational structure of the organization.	Yes.	TRUE
Are the organization's development teams capable of securing communications and validating in/outputs?	Yes, they try to facilitate this as much as possible.	TRUE
Are development teams aware of the risks caused by public interfaces?	Yes. They also have guard rails: endpoints are private unless requested otherwise.	TRUE
Are policies in place for adding new Serverless components to the IT landscape?	Yes.	TRUE
Are code based compliance and security checks imposed within the Ci/Cd pipeline?	Yes.	TRUE
Are automatic compliance and security validation checks imposed on the Cloud landscape?	Yes. They currently have alerts but no automatic reactions yet.	FALSE

### Financial Implications Analysis

*Analysis of business financials and investment requirements for new technologies*

Criteria	Comments	Response
Is the organization aware that Serverless is a long term investment, and is it capable of making this investment? I.e. costs will only drop after a long period.	Yes, but it depends on the business unit. Short-term business units are not always able to afford this.	TRUE
Can the organization invest when costs are difficult to asses? I.e. unpredictable cost fluctuations?	Yes, they are aware of what they call the bookmaker's nightmare;	TRUE
Does the organization's financial model allow an invocation based billing model?	Cloud itself was already a problem, Serverless even more. They are looking to solve this.	FALSE
Is the budget available for extra training and recruitment?	A considerable challenge because the market is very scarce.	FALSE
Is the budget available for extra governance?	No. They do not expect extra governance costs.	FALSE
Are quotas available for traffic costs during development by inexperienced development teams?	Yes. Teams are allowed and responsible for doing this themselves within their budget.	TRUE

### Cloud Deployment

*Cloud service provider partnerships and deployment*

Criteria	Comments	Response
Does the organization already use the database, API gateway, logging, monitoring, and security services the CSP provides?	Yes.	TRUE
Does the organization use testing, Ci/Cd, and version control tools?	Yes.	TRUE
Does the organization enforce Infrastructure as Code and Immutable Infrastructure?	Yes. The law forces us to do this.	TRUE
Does the organization have a distributed monitoring system in place?	Yes. They even have multiple.	TRUE

Table 11: Demonstration Questionnaire Details



Capgemini  invent

UNIVERSITY  
OF TWENTE.