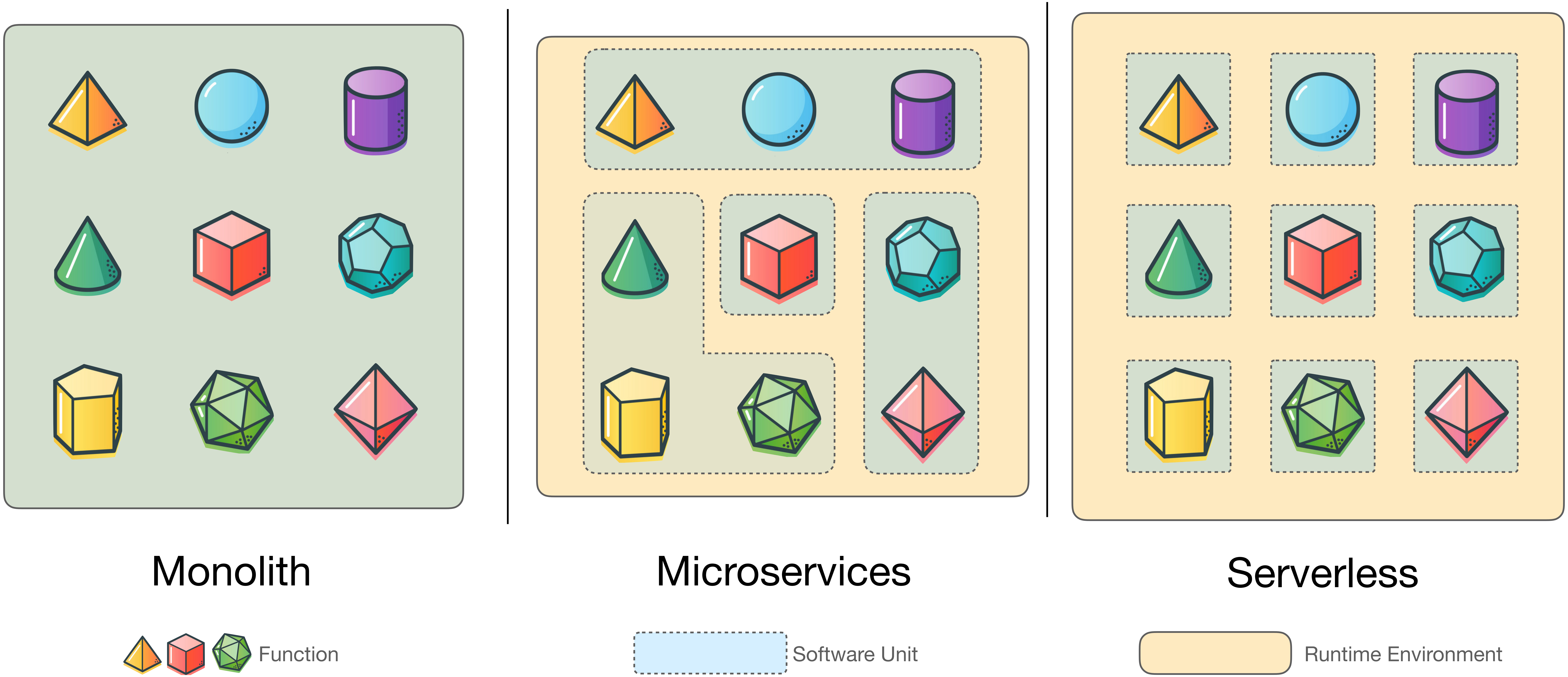
The Servers of Serverless Computing

A Formal Revisitation of Functions as Services

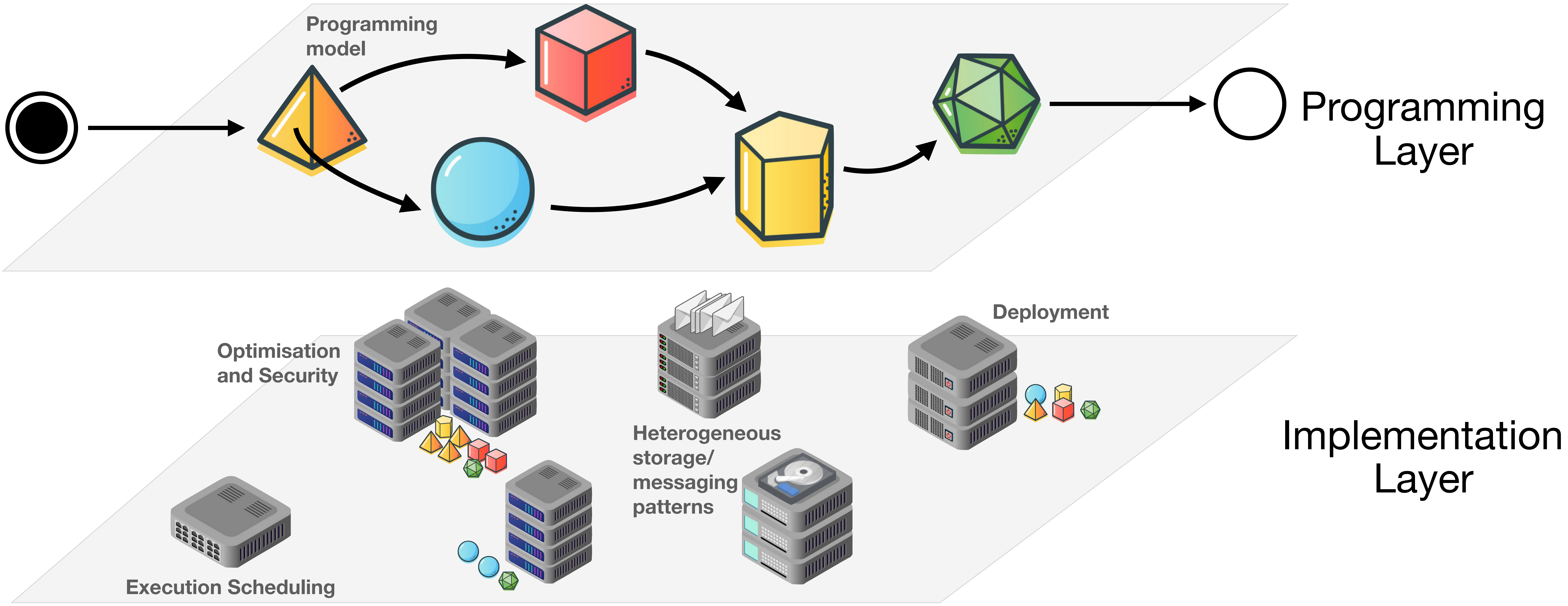
Saverio Giallorenzo[1,2former], Ivan Lanese[1], Fabrizio Montesi[2], Davide Sangiorgi[1], and Stefano Pio Zingaro[1]

[1]Università di Bologna/INRIA
[2]University of Southern Denmark

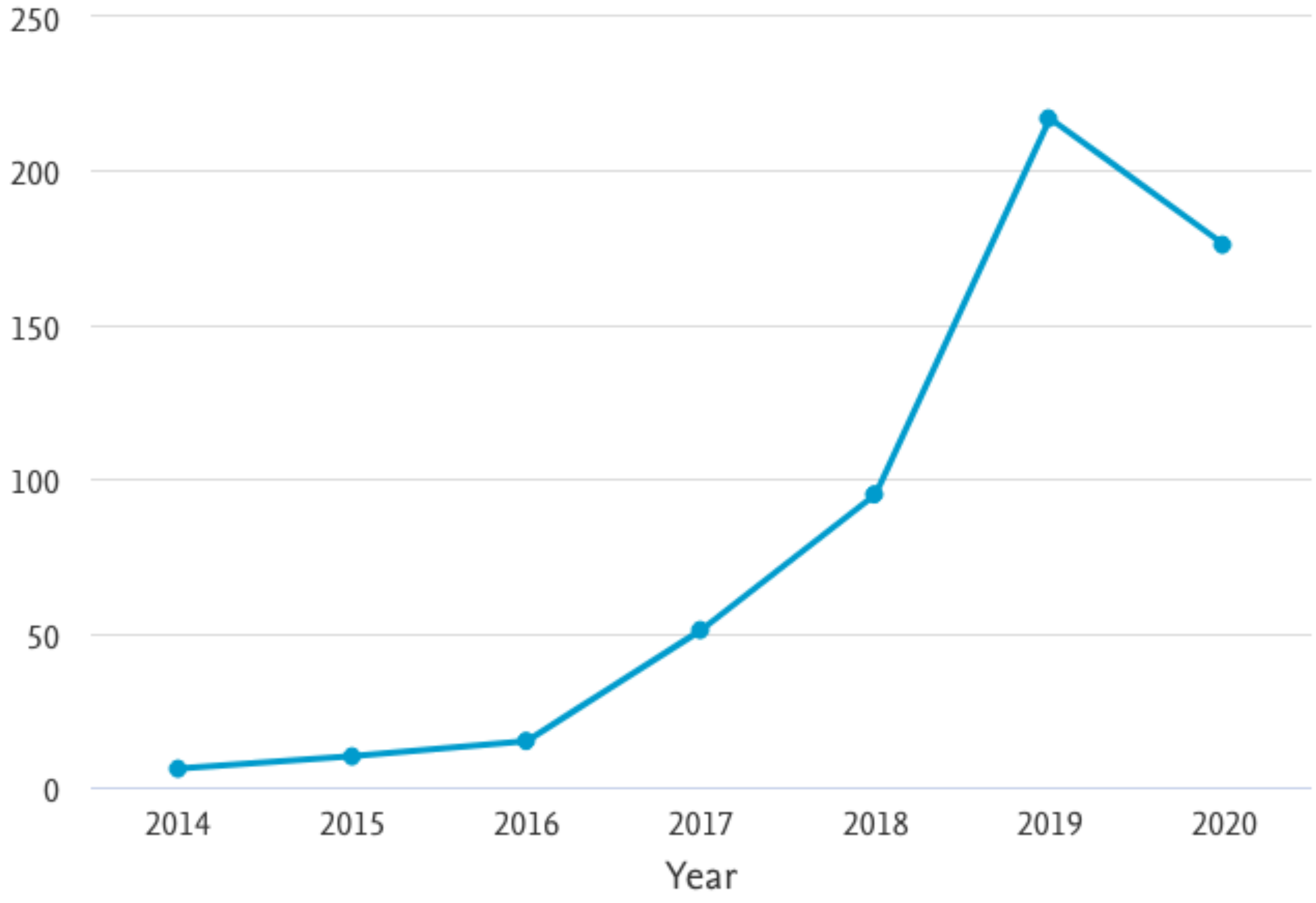
A Gentle Introduction to Serverless



A Gentle Introduction to Serverless



Serverless as Research Topic



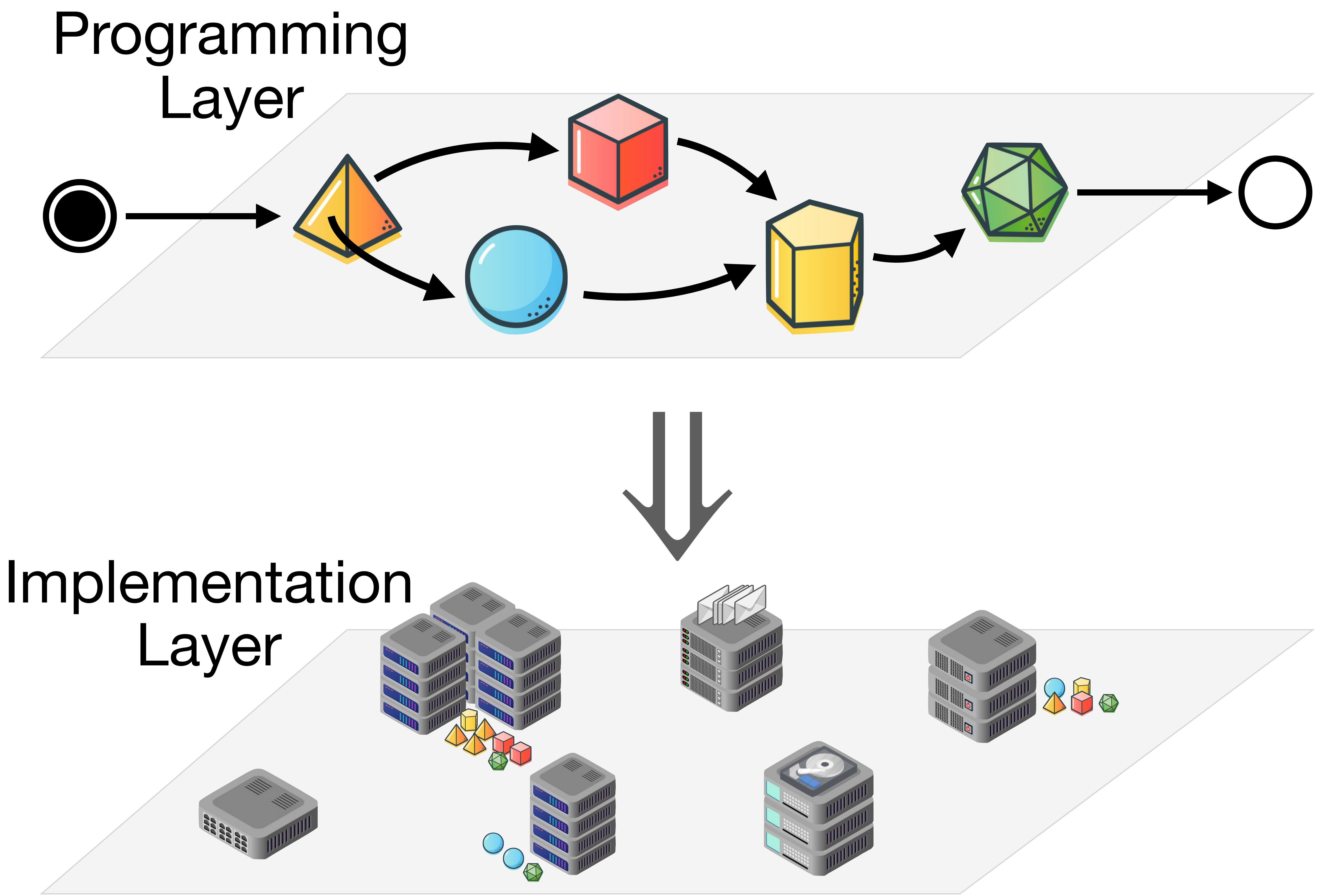
Source: Scopus

Serverless as Research Topic

Venue	# Papers	Core / SCIMAGO Rank
Future Generation Computer Systems	8	Software : Q1
IEEE Internet Computing	3	Computer Networks and Communications : Q1
IEEE Transactions on Parallel and Distributed Systems	2	Computational Theory and Mathematics: Q1
USENIX Annual Technical Conference + HotCloud	13 (6,7)	A / -
IC2E + IEEE CLOUD + CLOSER	20 (5,10,5)	- / B / -
ACM Symposium on Cloud Computing (SoCC)	12	-
SIGMOD	4	A*
Middleware	4	A
CIDR	3	A
OOPSLA	2	A*
ICSE	2	A*
INFOCOM	2	A*

Source: DBLP

The Servers of Serverless



SKC

influenced by λ and π calculus

$\Downarrow$

π calculus

SKC • Syntax

$$
\begin{array}{rlcl}
\text{Configurations} & C & ::= & \langle S, \mathcal{D} \rangle \mid \boldsymbol{\nu} n\, C \\
\text{Definition repository} & \mathcal{D} & ::= & \{(f_1, M_1), \ldots, (f_k, M_k)\} \quad (k \geq 0) \\
\text{Systems} & S, S' & ::= & c \blacktriangleleft M \mid S \shortmid S' \mid \boldsymbol{\nu} n\, S \mid 0 \\
\text{Functions} & M, N & ::= & M\ N \mid V \mid \\
 & & & \textsf{call}\ h \mid \textsf{store}\ h\ N\ M \mid \textsf{take}\ h \mid \boldsymbol{\nu} f\, M \mid \textsf{async}\ M \mid c \\
\text{Values} & V, V' & ::= & x \mid \lambda x.\, M \mid f \\
\text{Restrictable names} & n & ::= & c \mid f \\
 & h & ::= & f \mid x \\
\text{Function names} & f & \in & \textsf{Fun} \\
\text{Future names} & c & \in & \textsf{Fut} \\
\text{Variables} & x & \in & \textsf{Var}
\end{array}
$$

SKC · Simple Example

$$\langle c \blacktriangleleft \mathsf{call}\ f,\ D \cup \{(f, M)\} \rangle$$
$$\rightarrow \langle c \blacktriangleleft M,\ D \rangle$$
$$\rightarrow \langle c \blacktriangleleft V,\ D \rangle$$

SKC · Simple Example (async)

$$\langle c \blacktriangleleft \mathsf{async\ call}\ f, \mathcal{D}\rangle$$
$$\longrightarrow \langle \boldsymbol{\nu} c'\ (c \blacktriangleleft c' \mid c' \blacktriangleleft \mathsf{call}\ f), \mathcal{D}\rangle$$
$$\longrightarrow \langle \boldsymbol{\nu} c'\ (c \blacktriangleleft c' \mid c' \blacktriangleleft V), \mathcal{D}\rangle$$
$$\longrightarrow \langle \boldsymbol{\nu} c'\ (c \blacktriangleleft V \mid c' \blacktriangleleft V), \mathcal{D}\rangle$$

SKC • Example, Private State

$$(\; newLog, \underbrace{\nu log}_{\text{Fresh name /restriction}}(\text{store } \underbrace{log}_{\text{Name}} \; \underbrace{\text{call } \overbrace{nil}^{\text{Empty list}}}_{\text{Body}} \; \underbrace{log}_{\text{Continuation}}) \;) \in D$$

SKC • Example, Private State

$$(\ newLog, \nu log(\text{store } log \text{ call } nil \ log) \ ) \in D$$

SKC · Example, Private State

$$(\ newLog, \nu log(\text{store } log \text{ call } nil\ log)\ ) \in D$$

$$\langle c \blacktriangleleft (\lambda x\,.\,(\text{call } pair\ ((M\ x)(N\ x))\ x))\ \text{call } newLog, D \rangle$$

SKC • Example, Private State

$$(\ newLog, \nu log(\text{store}\ log\ \text{call}\ nil\ log)\ ) \in D$$

$$\langle c \blacktriangleleft (\lambda x\,.\,(\text{call}\ pair\ ((M\ x)(N\ x))\ x))\ \text{call}\ newLog, D\rangle$$

$$\langle c \blacktriangleleft (\lambda x\,.\,(\text{call}\ pair\ ((M\ x)(N\ x))\ x))\ \nu log(\text{store}\ log\ \text{call}\ nil\ log), D\rangle$$

SKC · Example, Private State

$$(\ newLog, \nu log(\text{store}\ log\ \text{call}\ nil\ log)\ ) \in D$$

$$\langle c \blacktriangleleft (\lambda x\,.\,(\text{call}\ pair\ ((M\ x)(N\ x))\ x))\ \text{call}\ newLog, D\rangle$$

$$\langle c \blacktriangleleft (\lambda x\,.\,(\text{call}\ pair\ ((M\ x)(N\ x))\ x))\ \nu log(\text{store}\ log\ \text{call}\ nil\ log), D\rangle$$

$$\nu log \langle c \blacktriangleleft (\lambda x\,.\,(\text{call}\ pair\ ((M\ x)(N\ x))\ x))\ log, D \cup \{(log, \text{call}\ nil)\}\rangle$$

SKC • Example, Private State

$$(\ newLog, \nu log(\text{store } log \text{ call } nil\ log)\ ) \in D$$

$$\langle c \blacktriangleleft (\lambda x\,.\,(\text{call } pair\ ((M\ x)(N\ x))\ x))\ \text{call } newLog, D \rangle$$

$$\langle c \blacktriangleleft (\lambda x\,.\,(\text{call } pair\ ((M\ x)(N\ x))\ x))\ \nu log(\text{store } log \text{ call } nil\ log), D \rangle$$

$$\nu log \langle c \blacktriangleleft (\lambda x\,.\,(\text{call } pair\ ((M\ x)(N\ x))\ x))\ log, D \cup \{(log, \text{call } nil)\} \rangle$$

$$\nu log \langle c \blacktriangleleft \text{call } pair\ ((M\ log)(N\ log))\ log, D \cup \{(log, \text{call } nil)\} \rangle$$

SKC • Example, Private State

$$(\ newLog, \nu log(\text{store}\ log\ \text{call}\ nil\ log)\ ) \in D$$

$$\langle c \blacktriangleleft (\lambda x\,.\,(\text{call}\ pair\ ((M\ x)(N\ x))\ x))\ \text{call}\ newLog, D\rangle$$

$$\langle c \blacktriangleleft (\lambda x\,.\,(\text{call}\ pair\ ((M\ x)(N\ x))\ x))\ \nu log(\text{store}\ log\ \text{call}\ nil\ log), D\rangle$$

$$\nu log\langle c \blacktriangleleft (\lambda x\,.\,(\text{call}\ pair\ ((M\ x)(N\ x))\ x))\ log, D \cup \{(log, \text{call}\ nil)\}\rangle$$

$$\nu log\langle c \blacktriangleleft \text{call}\ pair\ ((M\ log)(N\ log))\ log, D \cup \{(log, \text{call}\ nil)\}\rangle$$

$$\nu log\langle c \blacktriangleleft \text{call}\ pair\ (M\ log\ V_N)\ log, D \cup \{(log, N_{log})\}\rangle$$

SKC • Example, Private State

$$(\ newLog, \nu log(\text{store } log \text{ call } nil\ log)\ ) \in D$$

$$\langle c \blacktriangleleft (\lambda x\,.\,(\text{call } pair\ ((M\ x)(N\ x))\ x))\ \text{call } newLog, D\rangle$$

$$\langle c \blacktriangleleft (\lambda x\,.\,(\text{call } pair\ ((M\ x)(N\ x))\ x))\ \nu log(\text{store } log \text{ call } nil\ log), D\rangle$$

$$\nu log\langle c \blacktriangleleft (\lambda x\,.\,(\text{call } pair\ ((M\ x)(N\ x))\ x))\ log, D \cup \{(log, \text{call } nil)\}\rangle$$

$$\nu log\langle c \blacktriangleleft \text{call } pair\ ((M\ log)(N\ log))\ log, D \cup \{(log, \text{call } nil)\}\rangle$$

$$\nu log\langle c \blacktriangleleft \text{call } pair\ (M\ log\ V_N)\ log, D \cup \{(log, N_{log})\}\rangle$$

$$\nu log\langle c \blacktriangleleft \text{call } pair\ V_M\ log, D \cup \{(log, N_{log} :: M_{log})\}\rangle$$

SKC · Results, $SKC \leftrightarrow \pi$ Operational Correspondence

Theorem 1. From SKC-to-π operational correspondence

If $C \rightarrow C'$ then $[\![C]\!]^* \rightarrow \approx [\![C']\!]^*$

Theorem 2. From π-to-SKC operational correspondence.

If $\{[C]\}\ * \rightarrow P$ then there is C' with $C \rightarrow C'$ and $P \approx [\![C']\!]^*$

SKC • Future Work

- **guarantees** like *sequential execution*, *sequential consistency*, and *global-state transformation serialisability*;
- **programming models** that give programmers a global view of the overall logic of the distributed functions and capture the loosely-consistent execution model of Serverless;
- **transformation frameworks**, e.g., depending on the application context and inbound load, users/optimisation systems can transform parts of a given system from Serverless to Microservices and vice versa;
- **prediction models** for cost/resource usage, which require a modelling that relates functions and their execution at the implementation layer.

Thank for your time

Happy cruising!

Appendix

SKC • Example, applications and non-determinism

$$
\begin{aligned}
\langle\, & c_0 \blacktriangleleft \mathsf{store}\ wa\ (\mathsf{call}\ pair\ (\mathsf{call}\ cons\ 0\ \mathsf{call}\ cons\ 0\ \mathsf{call}\ nil)\ 1)\ ()\\
& \mid c_1 \blacktriangleleft \mathsf{call}\ trainAndStore\ wa\ (\mathsf{call}\ pair\ (\mathsf{call}\ cons\ 0\ \mathsf{call}\ cons\ 0\ \mathsf{call}\ nil)\ 0)\\
& \mid c_2 \blacktriangleleft \mathsf{call}\ trainAndStore\ wa\ (\mathsf{call}\ pair\ (\mathsf{call}\ cons\ 0\ \mathsf{call}\ cons\ 1\ \mathsf{call}\ nil)\ 0)\\
& \mid c_3 \blacktriangleleft \mathsf{call}\ trainAndStore\ wa\ (\mathsf{call}\ pair\ (\mathsf{call}\ cons\ 1\ \mathsf{call}\ cons\ 0\ \mathsf{call}\ nil)\ 0)\\
& \mid c_4 \blacktriangleleft \mathsf{call}\ trainAndStore\ wa\ (\mathsf{call}\ pair\ (\mathsf{call}\ cons\ 1\ \mathsf{call}\ cons\ 1\ \mathsf{call}\ nil)\ 1)\\
& \mid c_5 \blacktriangleleft \lambda\, w.\, (\mathsf{call}\ predict\ (\mathsf{call}\ cons\ 0\ \mathsf{call}\ cons\ 1\ \mathsf{call}\ nil)\ (\mathsf{call}\ first\ w)\\
& \qquad\qquad (\mathsf{call}\ second\ w))\ \mathsf{call}\ wa\,,\, D \,\rangle
\end{aligned}
$$

SKC • Example, applications and non-determinism

$$
\begin{aligned}
&\langle c_0 \blacktriangleleft \mathsf{store}\ wa\ \cdots \\
&\ \ \mid c_1 \blacktriangleleft \mathsf{call}\ trainAndStore\ \mathsf{call}\ wa\ \cdots \\
&\ \ \mid c_2 \blacktriangleleft \mathsf{call}\ trainAndStore\ \mathsf{call}\ wa\ \cdots \\
&\ \ \mid c_3 \blacktriangleleft \mathsf{call}\ trainAndStore\ \mathsf{call}\ wa\ \cdots \\
&\ \ \mid c_4 \blacktriangleleft \mathsf{call}\ trainAndStore\ \mathsf{call}\ wa\ \cdots \\
&\ \ \mid c_5 \blacktriangleleft \lambda w\,.\,(\mathsf{call}\ predict\ \cdots)\ \mathsf{call}\ wa, D\rangle
\end{aligned}
$$