Expression of Interest of future visions and research directions 2025-27 in the area of Cloud-to-Edge-to-IoT

H2020 PHYSICS (oPtimized HYbrid Space tIme Continuum in faaS) G.A. 101017047

Motivation

In the last decade, significant advancements have been performed in the way infrastructure and services are provided and provisioned. Infrastructure as Code techniques, cloud-oriented technologies for resource creation, management and optimization have evolved, as well the ability of these frameworks to scale.

On the other hand, application creation has become significantly more difficult, from one aspect due to this highly distributed and liquid environment. Dynamically created or adaptable resource environments create the need for aspects such as generalized parameterization of an application component, on the fly configuration and dynamic orchestration abilities from the application side as well to adapt to the execution substrate volatility. Distribution of application parts across the continuum, including IoT devices, edge, fog and cloud environments means that now significantly more risks ca occur during application operation, compared to a monolithic or even service oriented approach. The need for agility, speed and adaptability of an application has skyrocketed in order to keep up with the pace of developments and time to market pressures.

Current Status and Trends

What is more, an emerging gap in IT personnel numbers has been detected, indicating a shortage in the required numbers. For this reason, there is a generic trend in attracting people from non-tertiary ICT education[1] or from other domains (such as mathematics, physics and beyond) for IT positions or ones related to Data Science and AI. However these people are not highly skilled developers with extensive development background. So there is an urgent need to lower the abstraction of the development platforms as well as infrastructure as code and DevOps processes so that the latter can be taken advantage with a lower learning curve and technical background. Low-code development and operation environments therefore are a good candidate to undertake a more critical role.

The emergence of AI-based approaches such as Chat-GPT or GPT-4 may streamline a considerable size of the development work eventually, but this relates primarily to mainstream and typical processes and components. The lack of semantics and the ability to generate new knowledge from the AI realm in the foreseeable future means that humans will still be responsible for the main creative part of application combinatorial logic and/or integrations between systems in order to fulfil a more complex goal.

[1] https://ec.europa.eu/eurostat/statistics-explained/index.php?title=ICT_specialists_in_employment#General_developments_in_the_demand_for_ICT_specialists

At the same time, increased geopolitical tensions (like in the case of Ukraine) as well as differences on the viewed importance of data privacy and protection (like in the EU-US case) raise data sovereignty concerns as well as a careful consideration between openness and strategic autonomy. The legal framework in the EU, with the existence of the GDPR as well as with other advancements like the Schrems I&II case[2], may create a more sovereign environment for the citizen but it also creates significant complications for the application developers. Hence ways of automating compliance or compliance checking should be provided in a pattern-like manner in order to aid in creating data-sovereign applications from design time.

Research Challenges

Challenge 1: Pattern-based, low-code Application Development

From an application development and software engineering point of view, developers have been experiencing difficulties moving typical applications to cloud-native designs and this has become even more difficult in the case of hyper-distributed applications, in which considerations, configuration and operation needs to be performed across different locations and managing entities. Aspects such as extensive distribution of the components, latency/performance trade-offs, failures etc. that are more straightforward in centralized architectures now become daunting tasks for the developer. There is no single best solution for designing and implementing an application, given that itself it relies on the current conditions across the continuum and the dynamicity with which the application can react.

Design Patterns[3] are a very useful tool when it comes to designing and formulating common concepts, but their configuration is still a significant challenge. Typically a pattern may need to be parameterized and configured, thus a reasonable question is how a parameter is going to be set. This is further complicated by the fact that runtime conditions may dictate a different strategy in pattern setup from time to time, indicating that dynamic and adaptive pattern configuration needs to be applied. For this reason, they need to be accompanied by relevant AI/ML models, given the dynamicity of the latter to apply control logic that will optimize the pattern runtime, taking under consideration a number of factors, including application set KPIs, current conditions of execution (such as failures, load, specific application used, condition of external applications/services, legal compliance etc.). In order to be directly applicable by people that are not hard-core developers, the specific approach needs to be coupled with generalized implementation primitives that can be easily embedded and combined in new applications (e.g. through a low code, drag&drop visual environment) in order to abstract the learning curve and reduce development time. Therefore the 1st Challenge can be summarized as:

A top level, low-code, hyper-distributed application design framework, offering diverse design pattern implementations for application creation, setup and operation, dynamically configurable through embedded AI-driven logic, and covering diverse business models in which less or more entity integration is needed. The framework should aim for an abstracted app design and handling of common issues like performance, availability enhancement, legal

[2] https://www.dpa.gr/en/Organisations/Data_transfers_outside_EU/schrems_II
[3] Cloud Design Patterns: https://docs.microsoft.com/en-us/azure/architecture/patterns/

compliance, data collection and protection and mainstream functional aspects as ready-made logic blocks that can be combined to offer the final application structure.

Challenge 2: Pattern-based Platform Engineering Automation

From a Platform Engineering point of view and having to support a hyper-distributed application operation across the continuum, suitable automation mechanisms need to be in place, that can dynamically configure dependent distributed services as well as expose key abilities offered in a parametric manner. Such abilities include the support for diverse data sharing option primitives, platform services related to Infrastructure as Code pattern templates, DevOps processes etc. thus acting as a unifying layer for compute, data and code stacks and adding dynamic onboarding for new opportunistic cloud/edge resources. Therefore the 2nd challenge can be summarized as:

A mid-level platform services layer, aiming to achieve adapted system integration through resource setup and management patterns regulating data access and manipulation, identify management, federated constructs, event generation and distribution as well as dynamic orchestration across the continuum. This pattern-based view will apply the principles of liquid computing across the available platform layers while enabling reuse of the created resource templates. This will be in result expressed as more flexibility in implementing and orchestrating an opportunistic service network with fine-grained regulation and tradeoffs between performance, energy, locality and privacy among others.