Software Architecture for Distributed Edge Computing Systems

Urjaswala Vora[1]

[1]*Pennsylvania State University, University Park, State College, PA-16802, USA*

Abstract

The world is progressing towards the hyper-connected era where every object in our physical world will possess built-in computing and connectivity. As the number of these smart devices grows, these Systems of Systems will transition from integrated domain-specific solutions to horizontal platforms where all systems will be designed for knowledge-based automation, driving the decentralization of computing, communications and business processes. The recent unpredictability in various business process scenarios has high-lighted the need for collaboration and the advantages of participating. Edge computing facilitates segregation of computing responsibilities between edge devices and cloud infrastructure to process and analyze data. Researchers are working on reconfigurable edge computing systems based on varied technology solutions. We propose a software architectural framework to facilitate sustainability and evolvability in such ecosystems.

Keywords

Edge Computing, Distributed Systems, Precepts

1. Introduction

IoT envisions that most physical devices will be connected to communicate with data centers to exchange information. This introduces the next massive jump in scale of data production. The conventional centralized cloud computing is struggling to satisfy the QoS for many applications due to the rapid increase in the number of mobile devices. With 5G network technology on the horizon edge computing will become the key solution to solving this issue [11].

Edges of a network put strict requirements on packaging, delivering and deploying code on devices that are often costly and positioned at hard-to-reach remote locations. Code pushed to the edge thus needs to be self-contained, that is, each build must be complete within itself. Any change to the code requires a complete package deployment where the package includes code, configurations, required libraries and software-de-fined environments, ensuring the container or VM can run anywhere without any ex-ternal dependencies [1]. Latency has been always a key issue for edge applications, and hence it is important to keep as much functionality at the edge as possible. This tends to impact the application design as well as change management process significantly.

Real-time edge processing as well as server applications in the cloud are looking up to the possible acceleration by FPGA. Systems that need to change behavior at runtime and yet to be high-performing are the major candidates for such a software architecture. Thus, the significance of hardware reconfiguration and software rewriting is increasing. In this context, each of the technologies progressing individually along with the advancements in codesign have been proposed [6].

Another domain aiming to prioritize quality parameters such as robustness, adaptability, self-organization, flexibility, scalability and decentralization is Swarm Intelligence (SI) [12]. Zedadra et al. [12] propose SI-based framework of for complex system of systems of smart cities with edge-devices as swarm of entities. Swarm intelligence is the collective behavior of decentralized and self-organized systems / agents that work in continuous coordination.

ICSOB '22: 13th International Conference on Software Business, November 8–11, 2022, Bolzano, Italy
EMAIL: urja.vora@psu.edu (A. 1)
ORCID: 0000-0003-2895-7237 (A. 1)

In this paper, we aim to define an architectural framework for reconfigurable edge-computing based systems. We present a software design paradigm that facilitates re-configurability in edge-computing more effectively using technology advancements in FPGA and Swarm intelligence. Section 2 here discusses the related work done presenting the study of existing design methods used in edge computing softwares. Section 3 presents the precept as a design paradigm for accelerating reconfigurability in edge computing further. Section 4 concludes the paper.

2. Related Work

Reconfigurable computing intends to fill the gap between hardware and software, achieving potentially much higher performance than software, while maintaining a higher level of flexibility than hardware [5]. FPGAs and reconfigurable computing have shown to accelerate a variety of applications such as data encryption, which is able to leverage both parallelism and fine-grained data manipulation [5]. Ohkawa et al. [6] propose a method of system development which includes FPGA reconfiguration based on a context, which is defined in Context-Oriented Programming (COP). The crosscutting concern problem at runtime is claimed to be resolved. FPGA reconfiguration with software is then managed easily, though the time for switching context is majorly used for the reconfiguration of FPGA. The COP framework then manages re-programming timing and consistency with surrounding software systems.

Adário and Bampi[2] particularly discuss the pressing needs of powerful software tools to support the mapping of high-level language specifications into a runtime environment that automatically partitions the reconfigurable modules of the hardware [2]. Zhao, Xiao and Liu [13] present edge computing as a computing model between an edge-device and cloud computing centres and it enabling a wide deployment for IoT.

Zhao et al. propose a real-time reconfigurable edge computing system divided into cloud-based configuration information management module, high-performance embedded module based on FPGA and combinable function module based on multi-processor core. They do acknowledge that the research on the multi-level scheduling algorithm of cloud-edge collaboration is an open research problem [13].

Artificial Swarm Intelligence (ASI) facilitates decision-making by converging in unison on solutions that combine the participating agents' diverse knowledge, wisdom, insights, and opinions in real-time [7]. ASI combined with edge computing applications will provide the ultimate solution to the real-time analysis and processing of the data captured by the edge devices without any external dependency on the server-based or cloud-based infrastructure. Reconfigurability supported by FPGA will ensure the effective modification of configuration of the edge devices enhancing the QoS.

3. Precept Orientation for Reconfigurable Edge Computing

Shi et al.[8] propose the concept of computing stream to facilitate programmability in edge computing model. That is defined as a serial of functions / computing applied on the data along the data propagation path. The functions / computing could be entire or partial functionalities of an application, and the computing can occur anywhere on the path as long as the application defines where the computing should be conducted. We wish to position the design paradigm of precept in concurrence with the computing stream proposed for edge computing model and the concept of context-oriented programming based on which the dynamic reconfiguration of FPGA will result in reconfigurable edge computing architecture.

3.1. Precept

Precept is a declaratively programmed component that controls, and coordinates actions required to achieve a business goal of an activity. Precept implements the control structure of the activities using a rule set. The resulting architecture reduces control complexity and control coupling and hence is more modular and evolvable [10]. In a precept-oriented system, the activity is translated into a precept that comprises a set of declarative control flow rules. The rules control the execution of the modules. A precept’s execution cycle depends on the state change of its data parameters.

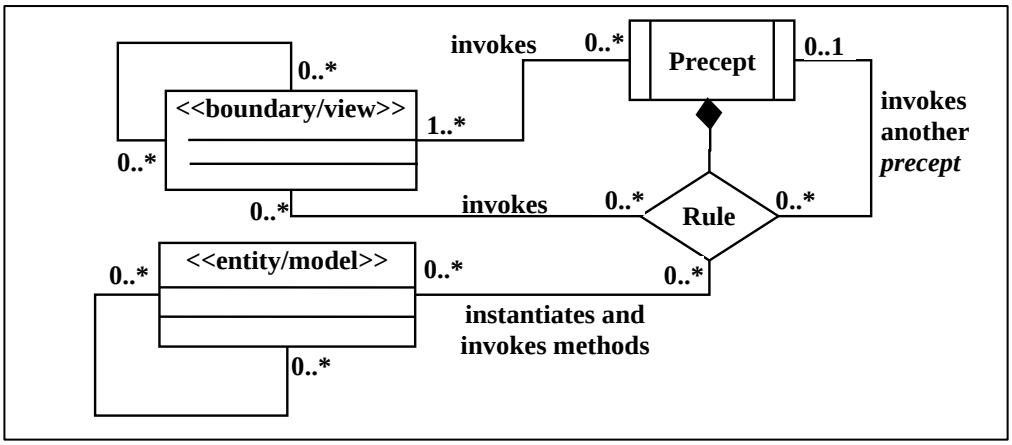

Figure 1: Precept Oriented Design

The precept orientated design is presented in Figure 1. The precept model is a critical element in the design of precept oriented system. A precept is defined by a 4-tuple as, Precept = ({input parameters}, {output parameters}, {internal parameters}, {rules}). A rule in precept oriented system belongs to a single precept. We define the rule by a 2-tuple as: rule = (pre-condition, action). Pre-condition defines the required state of the working memory for the selection of the rule. The working memory includes the input, output and internal parameters of the precept. The action of a rule contains the module(s) invocations. Precept oriented systems have an advantage that the rules as well as the data required of the rule selection and execution have a well-defined scope of the activity [10]. It makes a precept oriented system more maintainable than a rule-based system.

3.2. Precept-Oriented Design and Reconfigurable Architecture

FPGA is intrinsically a reconfigurable device which can be programmed as an arbitrary user circuit. The slow speed of reconfiguration is problematic at runtime reconfiguration. Partial reconfiguration partly solves this problem by reducing the size of bitstream. However, the time order of reconfiguration is much longer than the method switching of software [6].

Precept-oriented design is conceptualized to address the evolvability part of any system, especially for the ones having high control-flow complexity. We have illustrated with support of a number of case studies that control components in any system if are modelled similar to other imperative components, tend to become highly coupled and complex. Any changes to be made to such a system get propagated farther than anticipated when evolution impacts them. It was observed that control components, though approximately 1/4th in number of total components, bear more than 3/4th of the application's complexity making them high risk centres in the evolution of a software system [10]. A software architecture for ASI-based system requires the swarms of agents or edge computing applications to be collectively self-learning and self-modifying and work in collaboration to result in an intelligent system at the edges without any dependency on centralized components.

Precept oriented systems have control-flow modularised and externalised as rules for each activity leading to provide the reconfigurability desired by ASI-based edge computing and FPGA-based architecture. The computing stream proposed by Shi et al. [8] is software defined computing flow such that data can be processed in distributed and efficient fashion on data generating devices, edge nodes, and the cloud environment. The computing stream intends to help the user in determining the functions that need to be executed at edge and the data propagation thereafter.

When we try to provide for the reconfigurability in edge computing model with the help of technologies like ASI and FPGA, we need a design paradigm that will be able to determine high-risk centres or highly coupled components such as the control com-ponents of a software and segregate them and/or deal with them differently. This segregation can facilitate the resolution to deal with these components differently in time of reconfiguration of the software system. When this crosscutting concern of control-flow complexity within a software system is modularised declaratively it complements the philosophy of COP [6]. Precept-oriented design can be a strong solution for the

reconfigurability in edge computing model with its challenges such as the ones listed by Shi et al.[8], programmability, naming, data abstraction, service management, privacy and security, as well as optimization metrics.

4. Conclusion

The IoT paradigm raises a number of new challenges in the software engineering domain. To accomplish smart connectivity and context-aware computation, the Internet of Things demands software architectures and pervasive communication networks to process and convey the contextual information to where it is relevant [9]. Edge computing enables technologies allowing computation to be performed at the edge of the network, on downstream data on behalf of cloud services and upstream data on behalf of IoT services [8]. We have presented that IoT-based systems designed using precept-based architecture provide that balance of the controlled environment even though crowdsourcing is used for system's development and sustenance [9]. We extend the software architecture for reconfigurable edge computing model based on precepts and ASI which does concur with the resolutions offered for crosscutting concerns such as application of COP. Precept oriented design does aim to overcome challenges faced by cyber-physical systems that have explicit requirements of reconfigurable edge computing.

5. References

[1] State of the Edge 2021: A Market and Ecosystem Report for Edge Computing. The Linux Foundation, 2021.

[2] A. Adário, and S. Bampi: Reconfigurable Computing: Viable Applications and Trends. In Proceedings of the IFIP TC10/WG10.5 Tenth International Conference on Very Large Scale Integration: Systems on a Chip (VLSI'99), 1999, pp. 583–594.

[3] T. Batista, A. Joolia and G. Coulson: Managing Dynamic Reconfiguration in Component-Based Systems". In the Proceedings of the 2nd European conference on Software Architecture (EWSA'05), 2005, pp. 1–17.

[4] P. Chamoso, F. De la Prieta, F. De Paz and J. Corchado: Swarm Agent-Based Architecture Suitable for Internet of Things and Smartcities. In: , et al. Distributed Computing and Artificial Intelligence, 12th International Conference. Advances in Intelligent Systems and Computing, Vol 373. Springer, Cham, 2015.

[5] K. Compton and S. Hauck, S.: Reconfigurable computing: A Survey of Systems and Software. ACM Computing Surveys, Volume 34, Issue 2. 2002, pp. 171–210.

[6] T. Ohkawa, I. Tanigawa, M. Sato, K. Hisazumi, N. Ogura and H. Watanabe: Prototype of FPGA Dynamic Reconfiguration Based on Context-Oriented Programming. In Proceedings of 2019 IEEE 13th International Symposium on Embedded Multi-core/Many-core Systems-on-Chip (MCSoC), 2019, pp. 116-122.

[7] L. Rosenberg and G. Willcox: Artificial Swarm Intelligence. In the Proceedings of the IntelliSys 2019, UK.

[8] W. Shi, J. Cao, Q. Zhang. Y. Li and L. Xu, L.: Edge Computing: Vision and Challenges. In IEEE Internet of Things Journal, Vol. 3, no. 5, 2016, pp. 637-646.

[9] U. Vora et al.: Precept-Based Framework for Using Crowdsourcing in IoT-based Systems. In the Proceedings of 5th IEEE International Conference on Smart Computing, USA, 2019.

[10] U. Vora: Precepts and Evolvability of Complex Systems. In the Proceedings of the International Conference on Soft Computing and Software Engineering (SCSE), University of California, Berkeley, USA, 2015, pp. 565-574.

[11] W. Yu et al.: A Survey on the Edge Computing for the Internet of Things. in IEEE Access, Vol. 6, 2018, pp. 6900-6919.

[12] O. Zedadra, A. Guerrieri, N. Jouandeau, G. Spezzano, H. Seridi and G. Fortino: Swarm Intelligence and IoT-Based Smart Cities: A Review. In: Cicirelli, F., Guerrieri, A., Mastroianni, C., Spezzano, G., Vinci, A. (eds) The Internet of Things for Smart Ur-ban Ecosystems. Internet of Things. Springer, Cham, 2019.

[13] C. Zhao, C. Xiao and Y. Liu: A Real-time Reconfigurable Edge computing System in Industrial Internet of Things Based on FPGA. In Proceedings of 2021 IEEE 16th Conference on Industrial Electronics and Applications (ICIEA), 2021, pp. 480-485.