Behavior-Enabled IoT Architecture for Balancing QoE and QoS in IoT Systems

1st Assunta De Caro
Dept. of engineering
Università degli Studi del Sannio
Benevento, Italy
a.decaro6@studenti.unisannio.it

2nd Henry Muccini
DISIM Dept.
Università degli Studi dell'Aquila
L'Aquila, Italy
henry.muccini@univaq.it

3rd Eugenio Zimeo
Dept. of engineering
Università degli Studi del Sannio
Benevento, Italy
zimeo@unisannio.it

***Abstract*—This paper introduces Behavior-enabled IoT (BeT), a novel paradigm that pursues Quality of Experience (QoE) and Quality of Service (QoS) balance, through the establishment of a bi-casual connection between system and human agents. The system must adapt its performance or configuration to grant user experience, while human behavior can be influenced to reach QoS goals. The paper mainly focuses on the reference architecture and the modeling process.**

***Index Terms*—Software architecture, IoT, IoB, QoE, QoS**

I. Introduction

The ever-growing data stream from users' interactions with devices and technologies leads to new Internet of Things (IoT) paradigms, such as the Internet of Behaviors (IoB). These advancements shift the focus from simply connecting devices and collecting data to a deeper understanding and influence on human actions and behaviors. However, current IoB research often treats human and digital components separately, overlooking the critical trade-off between system performance (QoS) and user experience (QoE) [1]. While QoS expresses measurable and manageable service performance [2], QoE remains a multifaceted concept without a universally accepted definition in the literature. However, numerous studies acknowledge that QoE extends beyond performance parameters and highlight its dependence on a complex interplay of psychological, social, technological, and environmental factors, [3].

We propose a novel IoB-derived paradigm called Behavior-enabled IoT (BeT) [4]. BeT allows IoT software design that captures the bidirectional connection between humans and software, balancing these often conflicting needs. The system might autonomously adjust resources or reconfigure itself to maintain high QoE. However, influencing user behavior becomes necessary in critical situations with limited resources or high demand. This could involve suggesting actions or manipulating the environment to achieve desired outcomes.

BeT merges IoB, Internet of People (IoP) and Internet of Senses (IoS) aspects, to create a more comprehensive and user-centered IoB experience. IoP networks adapt to user movement through self-organizing architectures, allowing for dynamic resource allocation and service delivery. IoS leverages smart devices and edge intelligence to create richer user experiences by incorporating human senses like touch, smell, taste, and emotions. To date, an architectural paradigm that comprehensively combines and formalizes these concepts is missing. Therefore, this study fills the gap describing the BeT reference architecture and the related process for developing BeT concrete systems.

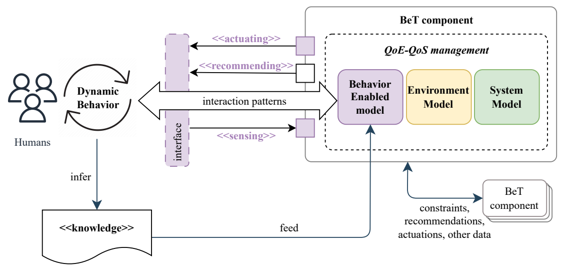

Fig. 1. BeT Reference Architecture

II. BeT reference architecture

We define a BeT system as a centralized or distributed software system with at least one BeT Component that processes user behavior data using a model to detect patterns. Capturing their interaction with IoT devices is necessary to interpret user behavior, therefore, IoT devices are part of the system's interface to the physical world. The collected data are ingested through the sensing interface. A Behavior Model (BeM) learns user patterns, by asking users for preferences, habits, and goals, or monitoring the alteration of the surrounding environment. An Environment Model (EM) reflects the evolution of environmental features not behavior-related. Lastly, a System Model (SyM) captures the system's configuration and computational resources.

This complex interplay of information is exploited by an adaptation strategy that optimizes both QoS and expected QoE. The internal processing of a BeT component is aimed at managing this balance. The behavior influence can be realized through context alteration (implicit mode) or suggestion (explicit mode). The first regards direct control of remote IoT devices, sending commands through the BeT Component's actuation interface; the second relies on indirect control, providing user recommendations. The behavior suggestion approach closes the loop leveraging on human awareness when

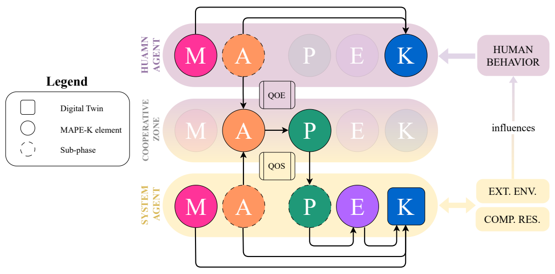

Fig. 2. BeT agents and roles

remote actuation is not feasible. Users' suggestions are forwarded through the recommendation interface. Furthermore, a BeT Component can receive exogenous supportive data from external modules or other sources. Multiple BeT components can collaborate, sharing responsibilities and data, and acting as a unified system.

BeT relies on an expanded concept of system adaptation. Unlike classical self-adaptive paradigms exclusively focusing on software adjustments, the BeT vision encompasses the physical environment and the human user's behavioral aspects. In self-adaptive systems, agents interact with each other and the environment using sensors and actuators to make informed decisions. MAPE-K loops (Monitor-Analyze-Plan-Execute) are a common way to implement this feedback loop for optimization. These loops consist of distributed agents that gather data (Monitor), decide on adaptation (Analyze), create an adaptation plan (Plan), and execute the plan (Execute). A common knowledge base is used for coordination. In a BeT system, two software agents balance QoE and QoS, and are mutually influenced by each other's actions, simultaneously playing the roles of manager and managed system.

- **Human-agent:** This agent focuses on the human-system interaction. It gathers data on user behavior (knowledge acquisition through monitoring) and then extracts information to build a behavior model (knowledge acquisition through analysis). This model is used to either predict expected QoE or define QoE constraints.
- **System-agent:** This agent monitors the surrounding environment and system resources. The constant changes in the knowledge we need to acquire may necessitate a digital twin to model this evolving environment. In addition, the twin actuates the requested changes in the real world. The analysis phase processes this data to estimate QoS or derive QoS constraints.

Both agents share an Analysis and Planning process. Agents work together in the Cooperative Zone to choose the strategy that maximizes QoE and QoS. The planning phase can result in several outputs:

1) **Explicit user behavior suggestion:** The system directly recommends an action for the user to take to satisfy a user's need or to reduce the system's resource demand peak.
2) **Environmental alteration to induce or support user behavior:** The system alters the environment to nudge the user towards a desired behavior.
3) **System performance adjustment:** The system modifies its configuration or scales computational resources to meet QoE/QoS requirements.

Strategies 2 and 3 are implemented and evaluated by the system agent's planning phase. However, recommendations can only be presented to the user for their decision. The user can accept or reject the suggested solution. Therefore, there's no execution phase on the human agent's side.

Fig.2 illustrates the relationships between the different phases in a BeT system, and highlights two key points:

- **Division of Labor:** While all MAPE-K steps (Monitor, Analyze, Plan, and Execute with a Knowledge) are crucial for BeT, not both agents need to perform every step.
- **Hierarchical Analysis and Planning:** The analysis and planning phases can be hierarchical. Analyses from both human and system sides can contribute to a shared analysis phase that balances QoE and QoS concerns. Similarly, system-side planning might be influenced by the solution proposed during shared planning.

III. BeT Modeling Process

Designing a BeT system according to the reference architecture sketched in Fig.1 requires different steps that can be grouped in three main phases. **1) QoE/QoS Analysis:** analyze stakeholders' needs to define QoE and QoS metrics and consider trade-offs and saturation points when setting acceptable ranges. Imagine a navigation app that prioritizes finding the shortest path (QoS). If many users choose the same shortest path, congestion can occur, leading to delays and frustration for everyone (poor QoE). The system might sacrifice a slightly longer route to avoid congested areas, resulting in a smoother journey for most users. **2) Behavior Interpretation and Influence:** distinguish observable properties to compute QoE/QoS metrics and controllable devices to collect them. Extract knowledge from data and design a recommendation strategy to influence user behavior for optimal QoE/QoS. **3) Concrete System Design:** define the BeT component responsible for implementing the recommendation strategy (AI-based model or simpler processor). Extend the framework with additional components specific to the application domain.

More details about the modeling process with examples of architecture concretization in smart city domains will be presented during the presentation at the conference.

References

[1] M. T. Moghaddam, H. Muccini, J. Dugdale, and M. B. Kjægaard, "Designing internet of behaviors systems," in *2022 IEEE 19th International Conference on Software Architecture (ICSA)*, Honolulu, HI, USA, 2022, pp. 124–134.

[2] B. Sabata, S. Chatterjee, M. Davis, J. Sydir, and T. Lawrence, "Taxonomy for qos specifications," in *Proceedings Third International Workshop on Object-Oriented Real-Time Dependable Systems*, 1997, pp. 100–107.

[3] K. U. Rehman Laghari and K. Connelly, "Toward total quality of experience: A qoe model in a communication ecosystem," *IEEE Communications Magazine*, vol. 50, no. 4, pp. 58–65, 2012.

[4] H. Muccini, B. Russo, and E. Zimeo, "The bet project: Behavior-enabled iot." *arXiv*, 2023.