Emergent Behaviors in Internet of Things: The Ultimate Ultra-Large-Scale System

Damian Roca, Daniel Nemirovsky, Mario Nemirovsky, Rodolfo Milito, and Mateo Valero

Abstract—The Internet of Things (IoT) promises a plethora of new services and applications. To reach its potential IoT must break down the silos that limit applications' interoperability and hinder their manageability. Doing so leads to the building of Ultra-Large Scale Systems (ULSS) in several verticals, including Autonomous Vehicles, Smart Cities, and Smart Grids. The scope of ULSS is large in the number of things and complex in the variety of applications, volume of data, and diversity of communication patterns. To handle this scale and complexity we propose Hierarchical Emergent Behaviors (HEB), a paradigm that builds on the concepts of emergent behavior and hierarchical organization. Rather than explicitly programming all possible decisions in the vast space of ULSS scenarios, HEB relies on the emergent behaviors induced by local rules at each level of the hierarchy. In this paper we discuss the modifications to classical IoT architectures required by HEB, as well as the new challenges. We also illustrate the HEB concepts in reference to Autonomous Vehicles. This use case paves the way to the discussion of new lines of research.

Index Terms—Internet of Things, Ultra Large Scale Systems, Emergent Behavior, Self-organization, Autonomous Vehicles.

1 Introduction

Internet of Things (IoT) includes a pervasive presence of sensors, actuators, and other devices that are deployed across large areas and connected via protocols (e.g. Bluetooth, WiFi, LoRA, 5G) that cooperate to meet common objectives [1]. The dominant characteristic of IoT is the physical interaction of the "things" with their environments, which enables novel applications and sets new architectural demands. Many of these applications are widely distributed, some have stringent real time requirements, and in all cases it is necessary to maintain trustworthy communication and adaptability to dynamic environments. IoT has the potential to significantly transform and improve city services, transportation, agriculture, health-care, energy production and distribution, and water conservation, among many other vital aspects of human life.

Conventional IoT deployments based on the simplistic approach of directly connecting "things" to the Cloud end up creating "silos" which limits the interoperability between applications. This approach complicates their orchestration and management, increases deployment costs, and it definitely does not support the scalability required to support autonomous vehicles, smart cities, transportation, and other relevant applications and services of interest. These applications are in fact Ultra Large Scale Systems (ULSS) [2], akin to the Systems of Systems (SoS) that, based on Maier's definition [3], are an assemblage of different components where each is both operationally and managerially independent. There are compelling reasons to decentralize ULSS. They include manageability, which complexity grows with the scale of the system; and the ability to contain failures.

To tackle the scale and complexity of these ULSS we propose Hierarchical Emergent Behaviors (HEB), a paradigm that fuses the fields of emergent behaviors and hierarchical organization. Emergent behavior, previously studied in numerous fields, including social behavior, biology, and ethology [4], has lately gained traction in robotics and autonomous vehicles [5]. The key idea is to induce self-organizing behaviors akin to the swarm formation, thus bypassing arduous, centralized, and potentially brittle control mechanisms.

While current IoT systems rely upon extensive coding to achieve an explicitly-programmed behavior, we propose imposing only minimal and lightweight local rules to regulate "things" interactions in order to achieve objectives through emergent behavior. Unlike swarms, the ULSS of interest operate at different levels in terms of space and time. This observation suggests the consideration of a hierarchical organization, in which level (N+1) abstracts the behavior of level (N) while widening its spatial-temporal scope.

This fusion of emergent behavior and hierarchical organization concepts enables inducing desired behaviors without the need to envisage nor explicitly program for the vast number of potential scenarios. A well-designed HEB promises to be more flexible and adaptable to unanticipated conditions than a traditionally hard coded system.

In this position paper, we aim to i) call the attention upon the scalability problem in IoT, ii) suggest an approach based on two well-known organizing principles, emergent behavior and hierarchical organization, and iii) stimulate future research based on the ideas and techniques related to Hierarchical Emergent Behaviors. To this end, we provide first an overview on how emergent behavior and the multidimensional approach applies to IoT. We then point out the architectural modifications needed in order to generate those behaviors and discuss some challenges. Finally, we

- *Damian Roca, Daniel Nemirovsky, and Mateo Valero are with the Barcelona Supercomputing Center (BSC-CNS) and the Universitat Politècnica de Catalunya (UPC), Barcelona, Spain. Email: {name}.{surname}@bsc.es*
- *Mario Nemirovsky is an ICREA Senior Research Professor at BSC-CNS.*
- *Rodolfo Milito is a Senior Technical Leader, Cisco Systems, Inc.*

illustrate its usefulness in dealing with an IoT ULSS through a case study.

2 Emergence in IoT

Emergent behavior may be defined as the collection of actions and patterns that result from local interactions between elements and their environment which have not been explicitly-programmed [6]. The local interactions themselves are driven by a set of engagement rules resulting in emergent behaviors and self-organization [7]. For instance, in his seminal paper Reynolds [8] defines flocking as the behavior that emerges when birds individually apply three local rules. The first rule seeks matching the speed of neighboring birds (alignment), the second rule prevents collisions with other birds or external objects by prescribing a minimum "bubble" around each bird (separation), and the third rule imposes a maximum distance between neighbors (cohesion). While none of these rules explicitly defines a collective behavior, a flock nevertheless emerges as a result of each individual bird flying according to its three given rules.

Collective behaviors such as flocks stem from the application of judiciously chosen local rules that are generic and independent of a specific time or location. The behaviors that emerge are a function of applying these rules within a particular environment (space, time, and contextual surroundings) which, when taken together, can be viewed as environmental constraints. Examples of such constraints are the quantity, density, and velocity of surrounding birds as well as the environmental obstacles such as walls. Applying these rules, however, does not necessarily lead to completely unpredictable behaviors. For instance, by knowing that the birds adhere to a collision avoidance rule, one can correctly predict that no two birds will crash even if they appear to be on a collision course with one another.

Transposing the flocking concepts to IoT results in viewing the "things" as the birds which follow given rules. As with the case of the birds, it is necessary to identify the constraints and choose appropriate rules that will affect the emergent behaviors for desired IoT-based services. For instance, Varaiya [9] laid the foundations to create groups of autonomous cars, called platoons. In his work a leader is designated in each platoon to manage group actions such as turns. These platoons can result out of an explicitly-programmed behavior (i.e. Varaiya) or emerge from a set of rules applied to each vehicle as the basic element taking also into account environmental constraints. The application of the three flocking rules to a set of autonomous cars, for instance, properly illustrates the concepts under discussion since the flock becomes a platoon of vehicles without the need of a leader.

Determining the constraints placed upon the "things" is a non-trivial feat that determines the balance between emergent and explicitly-programmed behaviors. Three factors compose the constraints imposed upon a "thing":

a) Capabilities: These are the functional attributes of an element. For example, an autonomous car may move in two dimensions, vary its speed, and measure pollution. The emergent behavior cannot extend beyond the limits imposed by the capabilities of the elements.

b) Rules: The rules govern the interaction of the elements with the environment and among themselves. What limits emergent behaviors is the strictness of the rules rather than their quantity. There are two types of hyper-parameters associated with the rules:

- Numerical parameters with a physical meaning (i.e., the separation distance to prevent collisions).
- "Weights" assigned to the rules in order to establish their precedence.

c) Environment: The numerous environmental factors at play can heavily influence how behaviors emerge in different settings. For instance, identical elements following similar rules may behave markedly different in a city compared to a jungle.

2.1 Decomposability and Hierarchical Emergent Behavior

The basic functionalities of a platoon are implicitly embedded in the local rules. The functionalities required of an ULSS, however, are more complex than that of a flock. Trying to define from scratch the rules required to enable the emergence of those behaviors would be a daunting task. Here, the pioneering work of Simon [10] on decomposability of large scale systems defines a hierarchy among subsystems that enables grappling with the complexity of a system effectively via an incremental approach. While the subsystems in Simon's work had explicit behaviors, we seek to combine this concept of decomposability with emergent behaviors in order to tackle the complexities of ULSS.

To this end, we propose extending the original emergent behavior approach by applying a hierarchical structure and tiered rules to the behaviors resulting from self-organization. Regarding IoT, Taft [11] proposes a hierarchical organization of smart grid. The focus is on the data aggregation at the different levels, with no concept of emergent behavior.

Our approach based on Hierarchical Emergent Behaviors (HEB) is depicted in Figure 1. The individual physical "things" are at the bottom of the hierarchy. A behavior emerges (e.g., the formation of a platoon) as a result of the application of the set of level 1 rules. The innovation behind the HEB concept lies in the application of a new set of rules to the emergent behavior resulting from the previous level. From the perspective of level 2, each platoon could be a level 2 "thing", which must follow the local level rules. For instance, level 2 local rules may establish certain minimum distance between platoons, and limit the number of vehicles within a platoon, from which the corresponding behavior emerges. While level 2 loses granularity due to aggregation, its scope in space and time is more expansive. The scheme is recursive, providing coarser granularity but wider scope as the hierarchy is climbed.

The HEB approach exploits the locality of interactions and perceptions since each hierarchical level provides a different vision of the elements. In the case of autonomous vehicles, for level 1 vehicles "locality" relates to the vehicles made "visible" through their on-line sensors. This "local" view does not include information about the way platoons are moving along the highway. The 2^{nd} level, rather than

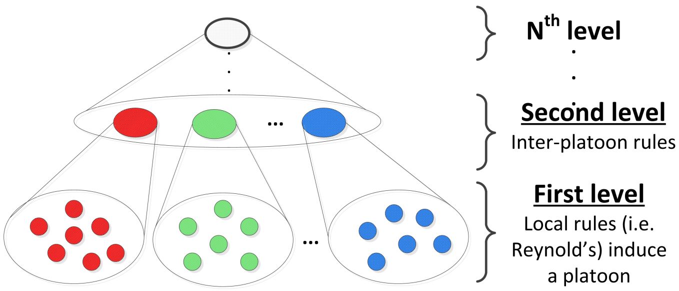


Fig. 1: Representation of the HEB concept with different hierarchy levels. The first level rules applied to level 1 elements (e.g. vehicles) induce a platoon behavior. The second level applies inter-level rules over the previous level behaviors (e.g. platoons) to enable more complex functionalities.

detailed information about the interactions within each platoon, keeps track of the flow of the platoons as single elements. This abstraction allows implementing different regimes of operation exploiting the "locality". For instance, during the normal regime the goal is to keep certain metrics (i.e. distance between them) at the desired level by tweaking the parameters that regulate the 1^{st} level behavior. The anomalous regime kicks in when, due to its long vision, the 2^{nd} level detects the onset of a congestion that requires rerouting.

2.2 Advantages of emergent behavior for IoT

The manner in which emergent behaviors cope with complexity and scalability issues is their main advantage. The functionalities as a system arise from local interactions and the element's flexibility instead of explicitly programming the uncountable number of possible situations.

This brittleness of the explicitly-programmed approach manifests itself even with systems composed of a few elements. The work by Saska et al. [12] used a specifically-programmed hawk-eyed supervising element that controlled and corrected the position of another set of autonomous vehicles. This central orchestrator is aware of the entire system and is responsible of maintaining the formation when faced with obstacles or failures. Consequently, it becomes the main bottleneck although the system is only composed of tens of elements. In comparison, emergent behavior eliminates the need of a central orchestrator that would have to deal with a very large number of "things". Moreover, the system's complexity is greatly reduced since decisions are taken in a distributed fashion, leveraging the intelligence of each "thing". Platoons emerge induced by the rules imposed over vehicles although each car is not aware of the size of the platoon nor are programmed to form groups.

Scalable IoT systems also need to adapt to the changes that occur in the dynamic environments where the "things" are deployed. Due to the large number of variables and situations, designing an explicit programmed system that takes into account all the scenarios in advance is a formidable task. With our HEB IoT approach and if the proper set of rules is defined, the "things" are able to dynamically adapt to the environment without the need to explicitly program them.

Lastly, applying HEB allows each higher level of the hierarchy to abstract away the complexities of the lower levels. This is the result of aggregating capabilities and data from individual "things", leading to less complex software development. For example, a level 2 "thing" (e.g. platoon) is composed of numerous level 1 "things" (e.g. vehicles). This process allows for a single query to be issued to a whole group instead of to each individual physical "thing".

3 Implementing Emergent Behavior for an ULSS IoT System

Current IoT architectures follow the structure depicted in 2. "Things" deployed for a single purpose are at the bottom of the architectural stack. These elements are sensors that measure their environment, and actuators that respond to commands. The connectivity layer manages the communication with the higher layers. Aggregation, curation, and in many situations processing (including deciding what must be handled locally, and what must be sent to the Cloud) are the main functionalities of the Data Layer. The application layer at the top defines and manages the tasks that compose the services.

3.1 Emergent Architectures

Implementing HEB on an IoT system requires modifications to this traditional application stack, impacting "things" and their communications.

In some IoT deployments "things" are passive sensors, and in many others they also include actuators. The actuation role gets heightened in HEB. In addition to their common operation regime, "things" will now take decisions based on their interactions with other "things" and the environment. For instance, the decision to join other vehicles in a platoon can be taken by the cars themselves without involving a higher orchestration layer. Hence, by 'active' we understand any device that participates in the generation of emergent behaviors or alters them (i.e. applying rules, modifying the environment, etc.). Distributing intelligence to the "things" exploits their locality while potentially reducing the managing complexity of an ULSS.

To sustain this new architectural feature, the communication capabilities require major changes. In a traditional design, architects must anticipate every possible scenario, and the communication patterns they entail. HEB's approach is radically different, in that the scenario space is explored using realistic simulations, but no claim is made that every single emergent behavior is covered. Rather, the effort is focused on ensuring the correctness of the local rules at the different levels of the hierarchy. Then, what matters in HEB is the interoperability between "things", achieved through standardized APIs and interfaces. Once "things" can communicate with each other and with their environment, new behaviors emerge by the application of the local rules. For instance, vehicles can be of different brands, and use a diverse array of sensors. Picture a vehicle communicating with other cars to join their platoon. If cars cannot communicate, the resultant behaviors would be very limited since not all the vehicles can become part of them. Instead, if the interoperability between cars is guaranteed, any vehicle could become part of a platoon. In addition, this vehicle could also communicate with sensors placed on streetlights to obtain contextual data that could improve the platoon's efficiency. Enabling these communications becomes critical to ensure the success of emergent IoT architectures.

A side effect of this communication can lead to an important advantage, the aggregation of data from heterogeneous sensors. Grouping sensors could lead to new functionalities without deploying new hardware. Imagine a platoon where each vehicle has a different set of sensors. Thanks to the abstraction provided by HEB, the platoon's capabilities will be the sum of each vehicle features. For instance, one vehicle measures the temperature and the following measures the pollution. In this case, the platoon can provide both measurements, hiding the fact that each vehicle only provides a single measurement. If the system aggregates sensors of the same type then the measurement's accuracy can improve by providing their average. In this case, a filter based on historical data and accuracy determines which sensor value is more likely to be correct.

The aforementioned changes translate into the addition of a new layer to the traditional IoT stack, the Rules layer, as shown in Figure 2. It is responsible for the rules and the hyper-parameters controlling the local interactions while maintaining the communication among "things". A different sub-layer handles each function. "Hierarchical Rules" sub-layer deals with the rules in each of the HEB levels and how they are applied over the physical "things". "Things communication" sub-layer manages the type of "thing", spatial and temporal information, security policies, and hierarchical queries and responses. The connectivity layer provides a channel to communicate resultant emergent behaviors with higher layers, while "Things communication" sub-layer focuses on the communication among "things" themselves to induce behaviors. Once a first level behavior is obtained, applying HEB converts the entire application stack into a second level "thing". In this case a new set of rules can be applied to the second level "things" in order to obtain more complex emergent behaviors. These behaviors end up in applications with more functionalities while maintaining the dynamism of the previous level "things".

3.2 Challenges

HEB holds tremendous potential to design and orchestrate ULSS, but the promise requires overcoming new challenges. Particularly critical challenges include: (i) behavior shaping, (ii) reliability, (iii) intra- and inter-level communications, and (iv) security.

3.2.1 Behavior Shaping

An intrinsic characteristic of emergent behaviors is that architects induce new behaviors by implementing different sets of rules [6] as well as by varying the complex environments. After applying a set of rules the architect performs a selection process to determine the useful behaviors. This decision becomes critical since it determines the application's functionalities. Non-useful behaviors are discarded and the rules adjusted correspondingly. Conversely, rules that are responsible for the useful behaviors can be enhanced and used more often.

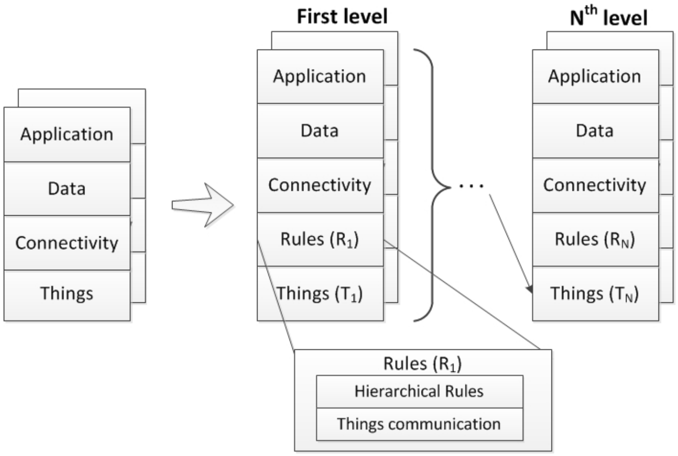


Fig. 2: HEB requires the layer "Rules" between the classical stack layers "Things" and "Connectivity", at every level of the hierarchy. For instance, if N=2, emergent behaviors out of level 1 become the level 2 "things". Then, the level 2 rules complement the level 1 rules.

Different optimizations can be applied to rules generating useful behaviors. One option consists of adjusting the rules' parameters to tune the behavior. For example, in a platoon we can adjust the collision avoidance distance so it covers a larger area. Another option is to add/remove rules to modify behaviors. The criteria to decide which behaviors are useful can be based on performance, formed groups, types of "things", and criticality among others. The result of this selection determines how effective the emergent IoT system will be.

3.2.2 Reliability

Provided that IoT is built upon millions of non-reliable low-cost sensors, we can expect a large number of failures aggravated by harsh environments. Fault propagation, similar to waves in water, should be stemmed. For example, Gerla et al. considered a platoon that suddenly stops for a crossing pedestrian [13]. Without proper boundaries, following platoons will also reduce their speed thus propagating the effect. A proper inter-platoon distance, controlled by 2^{nd} order rules, should avoid this problem.

A second problem arises when catastrophic failures occur due to bad behaviors. Instead of shutting down the entire system, there are two options. Either a functional backup is restored or a supervisor takes control of the system momentarily to manage the ULSS.

3.2.3 Intra- and Inter-level communication

Communication among "things" from different applications is a challenge by itself. Although identified years ago, it is not solved yet. HEB accentuates the urgency. For example, a platoon moves near another. To apply the second level rules (inter-platoon rules), it is necessary for both entities to communicate as level 2 "things". Open literature usually designates a leader to perform this task. However, centralization results in the loss of the locality and scalability advantages of HEB since a vehicle does not know the size of the platoon nor has membership awareness. Defining how entities from different levels communicate and how the rules interact will determine the performance of the behaviors.

3.2.4 Security

Emergent behaviors offer significant advantages in dealing with the overwhelming complexity of ULSS. However, they also extend the "attack surface" that can be exploited. An attacker that gains access could modify the rules, either directly or through modification of the hyper-parameters, for nefarious purposes. There is no magic bullet in security, but there are three major recommendations to follow: a) security is not add-on, incorporate it as an integral part of the design effort; b) leverage HEB context awareness to detect intrusions and other forms of attack; c) make sure that the design has the ability to isolate infections.

4 Case study

This section provides a case study to illustrate the potential real-world benefits that an emergent behavior approach has in IoT systems. Though our HEB should be applicable to any IoT system, here we focus on autonomous vehicles, as we have done throughout this article.

4.1 Fundamentals

A group of autonomous vehicles, either aerial (i.e. drones) or terrestrial (i.e. cars), constitute the basic elements of this system. For us, a vehicle is a sensor platform that applies rule-inducing behaviors as a single object, although it could be decomposed into its own component "things". Cars measure ambient temperature and pollution while drones focus only on pollution.

Each vehicle implements the three original rules from Reynolds (R_1 Alignment, R_2 Separation, and R_3 Cohesion). In addition, they have a rule to reach a target destination point (R_4 Destination) and a level 2 rule to induce a platoon of platoons (R_5 Second). To highlight the complexity of each rule, R_5 performs the same operations as R_1, R_2, and R_3 combined but applied to level 2 "things". Moreover, each rule is weighted by a value that can be modified (or even deactivated) in real time to observe their impact on the induced behaviors.

With these set of rules, the vehicles are ready to circulate in a city. This is a dynamic environment with limitations to their mobility (streets and obstacles) and other working IoT systems that translate into a huge amount of possible interactions and situations.

4.2 Methodology

We chose the Processing simulator [14] to perform our analysis of flocks. We modified the base flock example to add the new rules and constraints. As a consequence, each vehicle determines its trajectory based on the five implemented rules guided by its local interactions. Each type of vehicle is represented with a different color that also indicates its mobility patters (i.e. drones can fly over obstacles) and R_2 is modified so cars avoid their surrounding obstacles as well. To represent streets and other orographic patterns, we use obstacles shown as black dots in the canvas.

With these modifications, Processing offers a rich framework to simulate and visually observe vehicles, their environment, and their interactions. We also implement a mechanism to detect incorrect behaviors due to the violation of the rules (i.e. a car passing through an obstacle), complemented by the visual validation using the simulation canvas. The overall performance of the system is evaluated using both the visual observation and the actual alarms.

4.3 Emergent Autonomous Vehicles

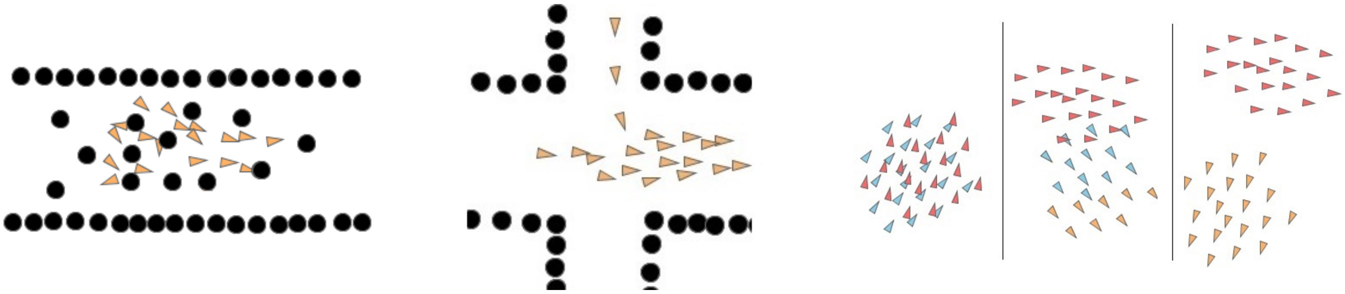
The simulation begins when we place a set of autonomous cars on a side of a straight street and specify the destination (through R_4). After a certain time, a platoon of cars is formed induced by R_1, R_2, R_3. For now, R_5 is deactivated (weight = 0) to focus on the level 1 behaviors.

Circulating as a platoon, the vehicles face a part of the street full of obstacles (i.e. non-emergent vehicles). At this moment the platoon behavior dissolves because each car focuses on avoiding obstacles (R_2 prevails over the rest for safety), as reflected in Figure 3a. This illustrates the effect of a changing environment on behaviors. Here cars were the moving element although it could have been reversed. Consider, for example, an IoT system of deployed sensors on lampposts to monitor certain traffic patterns. Therefore, "things" are static but they react to mobile elements in the environment (i.e. cars).

Once cars overcome the obstacles, their self-organization again results in the formation of a platoon. Next, we modify the separation distance in R_3 to observe the impact the rule's parameters have on the behaviors. Increasing this parameter results in the dispersion of the platoon, which may be desirable in search and rescue applications, for example. On the other hand, inadequately tuning these parameters may result in no or little constructive emergent behaviors. This shows that changes in the rules' hyper-parameters (weights and numerical values) greatly affect behaviors. In consequence, it may be desirable to adjust them in real time to efficiently deal with the dynamism of the ULSS.

The platoon now faces an intersection with another platoon on the road to its left. Since R_5 is still deactivated, they interact as level 1 "things" and not as platoons. R_4 determines whether the two platoons will form a larger grouping or each will continue their way, depending on their destination targets. In any case, the local interactions among cars prevent any collisions thanks to R_2. Since they have the same destination, they indeed form a larger platoon and continue advancing, as shown in Figure 3b. Following these rules, there was no need for a traffic light system to act as a global orchestrator between vehicle traffic.

Finally, the platoon reaches the end of the street and faces an unbounded area (no street delimiter). This platoon encounters a flock of drones induced by R_1, R_2, R_3. Although level 2 rules can apply among platoons of the same "things", we used aerial autonomous vehicles (drones). If we now activate R_5, then when the flock and the platoon are close enough, they will form a second level group, inducing new emergent behaviors. The cars in the platoon now use the capabilities of the drones inside the flock to scout the optimum route to avoid traffic and obstacles. Conversely, the drones in the flock may use the street level sensor data to



(a) Emergent behavior shaped by the street and obstacles. The platoon does not emerge due to the environment.

(b) Two platoons approaching an intersection without an orchestrator (traffic lights). In this case they form a larger platoon.

(c) Sequence of interaction between a flock of drones and a platoon of cars based on second level rules.

Fig. 3: A simulated platoon faces different situations while moving in a city environment.

increase their pollution measurement accuracy. Their level 1 functionalities are still preserved even though now they share information as level 2 “things”.

Figure 3c shows this situation. Cars (in red) and drones (in orange) can overlap since drones fly over them. The drones forming part of the level 2 group change their color to blue to indicate this new service. Not all the drones are part of the second level group due to the physical separation with the cars, which adds flexibility to HEBs. Once this distance becomes too large they split off, returning to level 1 behaviors only. Then, R_5 enabled the interoperability of different applications to provide new functionalities as a system.

5 Future Work

We have outlined a program to tackle the ULSS IoT challenge, opening four key areas for future research. The first line of research focuses on extensive realistic simulations of well-designed scenarios as a design tool. The emulation is the experimental design platform that allows architects to determine which rules are more likely to produce interesting behaviors, tune their hyper-parameters, and assess the performance in dynamic environments.

Machine Learning (ML) of rules that generate emergent behaviors is the second line of research. We note the strong link between the emergent behavior approach and ML, in both circumventing the need of explicit programming. ML can become valuable in choosing and tuning the hyper-parameters of local rules, and even identifying new useful rules. The experimental platform randomly sweeps the scenario space, and for each scenario ML tunes the hyper-parameters, or otherwise introduces new rules.

The last suggested research line involves security and reliability. Emergent behaviors bring new strengths: the system that can adapt itself as a whole in the face of new circumstances, including failures. On the flip side, there are new vulnerabilities to overcome, such as those that can be introduced through rogue “things”, or by altering the hyper-parameters of the local rules.

6 Conclusions

Many IoT services and applications are intrinsically large and complex, fully qualifying as ULSS. In this paper we address the problem of architecting and orchestrating ULSS. We propose a Hierarchical Emergent Behaviors (HEB) approach that borrows concepts from the fields of emergent behaviors and hierarchical decomposition.

By fusing emergent behaviors and hierarchical organization concepts, our architecture (HEB) uses only a minimal sets of engagement rules to achieve the desired behaviors without the need to explicitly program for every scenario. These techniques provide the system with less developer complexity and a natural ability to scale and adapt in dynamic environments. In order to implement this approach, we added a new layer to the traditional IoT architectural stack that incorporates the local rules of engagement. The autonomous vehicles case study was provided to illustrate the main properties and behaviors of “things” using the HEB approach.

Acknowledgments

Damian Roca work was supported by a Doctoral Scholarship provided by Fundación La Caixa. This work has been supported by the Spanish Government (Severo Ochoa grants SEV2015-0493) and by the Spanish Ministry of Science and Innovation (contracts TIN2015-65316-P).

References

[1] L. Atzori, A. Iera, and G. Morabito, “The internet of things: A survey,” *Computer networks*, 2010.

[2] L. Northrop, P. Feiler, R. P. Gabriel, J. Goodenough, R. Linger, T. Longstaff, R. Kazman, M. Klein, D. Schmidt, K. Sullivan *et al.*, “Ultra-large-scale systems: The software challenge of the future,” DTIC Document, Tech. Rep., 2006.

[3] M. W. Maier, “Architecting principles for systems-of-systems,” in *INCOSE International Symposium*. Wiley Online Library, 1996.

[4] H. J. Jensen, *Self-organized criticality: emergent complex behavior in physical and biological systems*. Cambridge university press, 1998.

[5] A. Kushleyev, D. Mellinger, C. Powers, and V. Kumar, “Towards a swarm of agile micro quadrotors,” *Autonomous Robots*, 2013.

[6] M. J. Mataric, “Designing emergent behaviors: From local interactions to collective intelligence,” in *Proceedings of the Second International Conference on Simulation of Adaptive Behavior*, 1993.

[7] E. Bonabeau, G. Theraulaz, J.-L. Deneubourg, S. Aron, and S. Camazine, “Self-organization in social insects,” *Trends in Ecology & Evolution*, 1997.

[8] C. W. Reynolds, “Flocks, herds and schools: A distributed behavioral model,” in *ACM SIGGRAPH computer graphics*. ACM, 1987.

[9] P. Varaiya, “Smart cars on smart roads: problems of control,” *Automatic Control, IEEE Transactions on*, 1993.

[10] H. A. Simon, *The architecture of complexity*. Springer, 1991.
[11] J. Taft and P. De Martini, "Ultra large-scale power system control architecture," *Cisco Systems*, 2012.
[12] M. Saska, V. Vonásek, T. Krajník, and L. Přeučil, "Coordination and navigation of heterogeneous mav–ugv formations localized by a hawk-eye-like approach under a model predictive control scheme," *The International Journal of Robotics Research*, 2014.
[13] M. Gerla, E.-K. Lee, G. Pau, and U. Lee, "Internet of vehicles: From intelligent grid to autonomous cars and vehicular clouds," in *2014 IEEE World Forum on Internet of Things*,.
[14] Processing simulation framework. [Online]. Available: https://processing.org/

Damian Roca is a PhD candidate at the Universitat Politècnica de Catalunya and a researcher in the Unconventional Computer Architectures and Networks Group at the Barcelona Supercomputing Center. His research interests include the Internet of Things (IoT), Fog Computing, novel simulation techniques, and computer architectures. Roca has an MSc in innovation and research in informatics with specialization in high-performance computing from the Universitat Politècnica de Catalunya. Contact him at damian.roca@bsc.es

Daniel Nemirovsky is a PhD candidate at the Universitat Politècnica de Catalunya and a researcher in the Computer Architecture for Parallel Paradigms Group at the Barcelona Supercomputing Center. His research interests include high-level language architectures, cache hierarchies, and scheduling techniques on heterogeneous chip multiprocessors. Nemirovsky has an MSc in computer architecture, networks, and systems from the Universitat Politècnica de Catalunya. Contact him at daniel.nemirovsky@bsc.es

Mario Nemirovsky is a Catalan Institution for Research and Advanced Studies (ICREA) Senior Research Professor at the Barcelona Supercomputing Center. His research interests include computer architectures, high-performance computing, the Internet of Things, and emerging on-chip interconnect technologies. Nemirovsky has a PhD from the University of California, Santa Barbara. Contact him at mario.nemirovsky@bsc.es.

Rodolfo Milito, a senior Technical Leader with the Chief Technology and Architecture Office of Cisco Systems, is currently engaged in IoT, Big Data and Machine Learning, and working on a distributed compute, storage, and network platform from the edge to the core of the network nicknamed "Fog Computing". Rodolfo got his PhD in EE (Stochastic Control Systems) from UIUC. Contact him at romilito@cisco.com.

Mateo Valero is a full professor at Computer Architecture Department at the Universitat Politècnica de Catalunya and the Director of the Barcelona Supercomputing Center. His research focuses on high-performance computer architectures. His main awards: Seymour Cray, Eckert-Mauchly, Harry Goode, ACM Distinguished Service. He is Honorary Doctorate by 7 Universities. He is a fellow of IEEE and ACM and is and Intel Distinguished Research Fellow. He is member of 5 academies. Contact him at mateo.valero@bsc.es.