Accepted to IEEE sponsored IntelliSys 2019, 5-6 September 2019 – London, United Kingdom

Containerized Autonomous Agents for Cognitive and IoT Applications

Tammy R. Fuller[1] and Gerald E. Deane[2]

[1] Chief Architect/VP, Echo Messaging Systems, Inc., 198 Chapel Street, Lincoln, Rhode Island 02865, USA
tammy@echomessaging.com
[2] CEO/President, Echo Messaging Systems, Inc., 198 Chapel Street, Lincoln, Rhode Island 02865, USA
gerald@echomessaging.com

Abstract. Multi-agent autonomous systems solve many problems by organizing processing actions around triggering events. Agents working independently of one another, target their own specific criteria based on data, time, location or any combination of these. Responses to triggered events result in new or updated data, notifications or actions upon processes. Virtualizing the processing environment of agents decouples computing requirements and resources from specific agent functions to provide better resource management, expanded agent actions, and more sophisticated adaptation algorithms, included those based on machine leaning (ML) and deep learning (DL). Applications based on containerized autonomous agents where ML and DL based adaptations, and computing resources managed by container orchestration, are more dynamic, responsive and adaptive, where agents can number in the tens or hundreds of thousands or more. Structured data from unstructured data sources are the basis for cognitive and Internet of Things (IoT) applications. Intelligent systems with containerized autonomous agents are well suited to these types of applications. This paper describes the theory and applied real-world use of containerized autonomous agents in intelligent systems.

Keywords: Agents, Autonomous Agents, Intelligent System, Machine Learning, ML, Deep Learning, DL, Ambient Intelligence, Artificial Intelligence, Containers, Smart Containers, IoT, Internet of Things, Antivirus, Cyber security, bots, Cognitive applications, anomaly detection, multi-agent, unstructured data, structured data, big data, Agent containers, AI containers, micro services

1 Introduction

Applying AI theory in real world environments enhances research findings because real world problems, even basic ones, are always more complex when everyday challenges of incomplete data, improper usage, and corrupt deployment are part of all typical scenarios. AI is well suited to dealing with these real-world problems and often look to human intelligence for guidance on how to automate cognitive processes such as extracting structure from the myriad unstructured data sources available on the Internet.

When commercial problems aren't solved by currently available solutions, customers are willing to accept any type of solution as long as the problem is solved reliably and cost-effectively, regardless of it involving artificial intelligent, machine learning, or multi-agent based systems. Our research into intelligent systems using autonomous agents is based on solving problems of automation common to small and medium businesses, particularly those that are widespread, such as automated dynamic personalized workflows, or building structured data from unstructured data sources.

Towards this effort of providing commercial solutions to currently unsolved or poorly solved problems of automation, we developed an AI-based framework architecture with a reusable agent library for an intelligent system with autonomous agent called ADIN, for "Anomaly Detection and Intelligent Notification" [1].

1.1 Background

At a foundational level, autonomous agents are processing units that have three key components: triggering criteria, action response and adaptation. Triggering criteria define membership into a group that is acted upon by the associated action response. Once the action is completed, metadata about the action is stored and are analyzed by other 'health' agents that seek out and anticipate normal patterns and alert to abnormal anomalies.

Agents in the ADIN intelligent system are constructed from a library of reusable triggers, actions and adaptations and a REST API provides a simple way to programmatically create, and run agents to solve larger and more complex problems involving data, time, space or any combination. As shown in Fig 1, agents are combined from simpler agents (1,2) to trigger from more complex definitions (3,4) based on Data, Time, Location or any combination. Entire systems of ADIN cells based on agents solve more and more complex problems

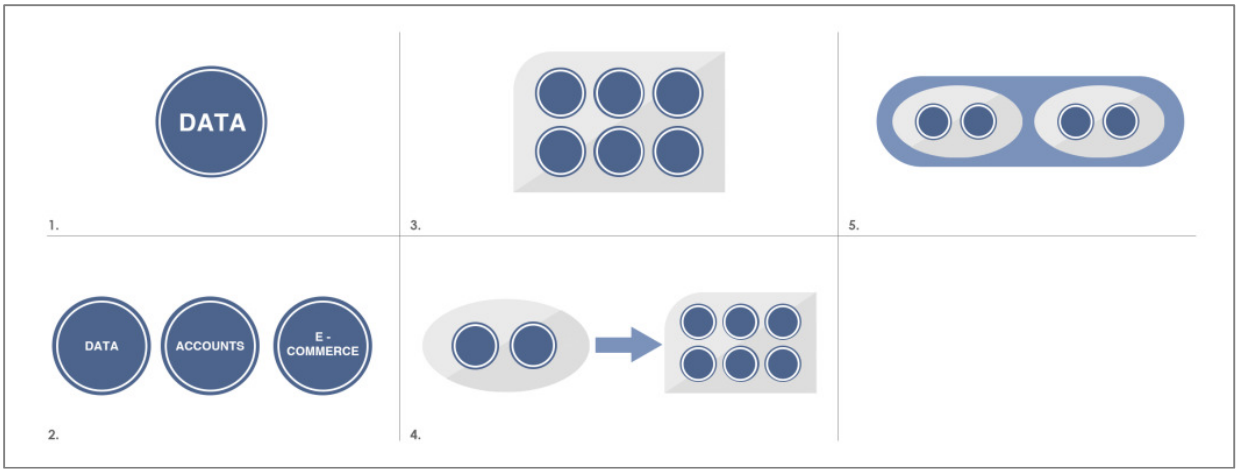

Fig. 1. Agents are combined from simpler agents (1,2) to trigger from more complex definitions (3,4) based on Data, Time, Location or any combination. Entire systems of ADIN cells based on agents solve more and more complex problems

Starting with basic triggering criteria, agents respond to simple events, as shown by examples in Table 1. By combining and aggregating agents, the ADIN Platform is commercially used for automated notification systems, emergency safety applications, data syncs, geospatial applications, Internet of Things (IoT) applications, interactive dashboards and many more as are described in later sections.

Table 1. ADIN Agents – Basic Triggering Examples

A new customer registered for an account
10 minutes have elapsed after a resource has been requested
A GPS enabled device indicates it crossed into a predefined Geo-fenced region

Agent triggers monitor data for new data, updated data, a specific field value, or deleted data. Time based triggers involving checking at a particular time of day, week, month, checking regular frequency intervals such as once per hour, or every 5 minutes. Likewise, location based triggers involving checking if a location enabled device crosses into or out of a predefined Geo-fenced region or line, or at frequency intervals such as every 100 meters.

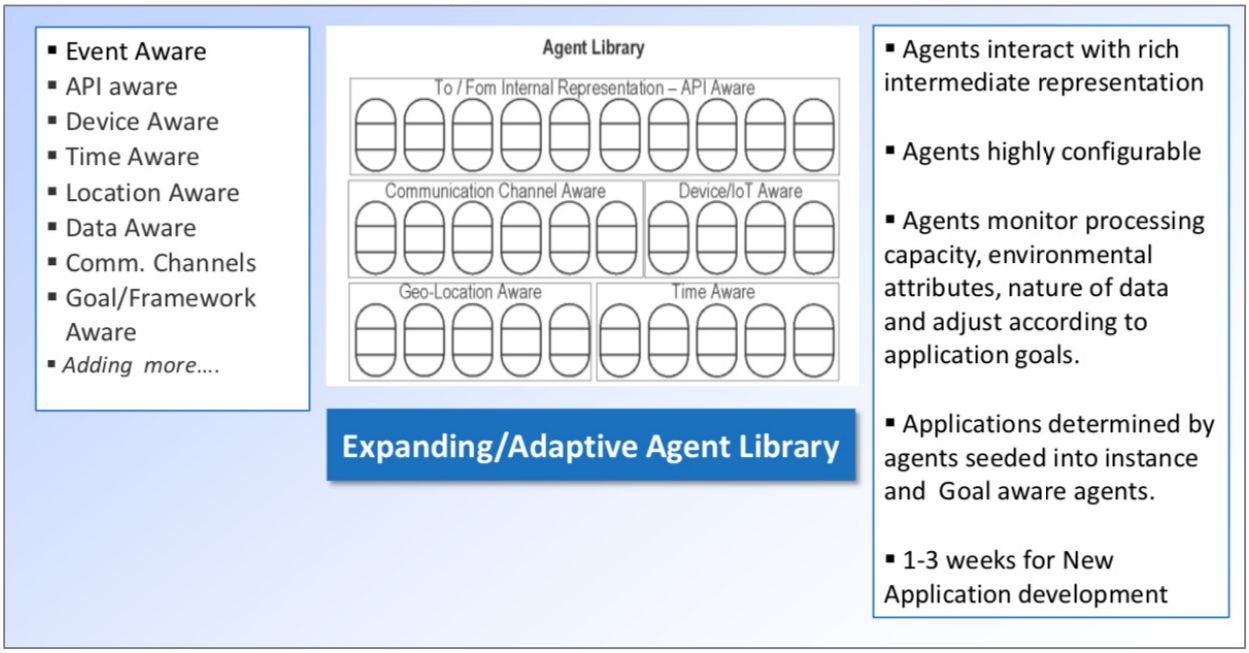

Fig. 2. Reusable agent library hosts building blocks to more complex solutions solved using autonomous in an intelligent system

Trigger definitions are combined using set theory operations, such as union, intersect, or complement to define any combination of Data, Time and or Location based triggers, as show in Table 2:

Table 2. Compound Triggering Examples

A new job is request was created 15 minutes ago with not status
1 mile is detected between two GPS enabled devices
10 minutes have elapsed after a resource has been requested
A GPS device crossed into a predefined Geo-fenced region at noon on Tuesday.
An API indicates a warning status on an IoT device between midnight and 4am

1.2 Agent components

Autonomous agents working in concert with one another from common data sources, being acted upon by users via common user interfaces are organized into units call ADIN Cells. Figure 3 shows a dashboard that monitors all applications using agent-based processes. Visualization allows for

quick human interaction of agents and ADIN cells processing normally vs. those that have experienced anomalies.

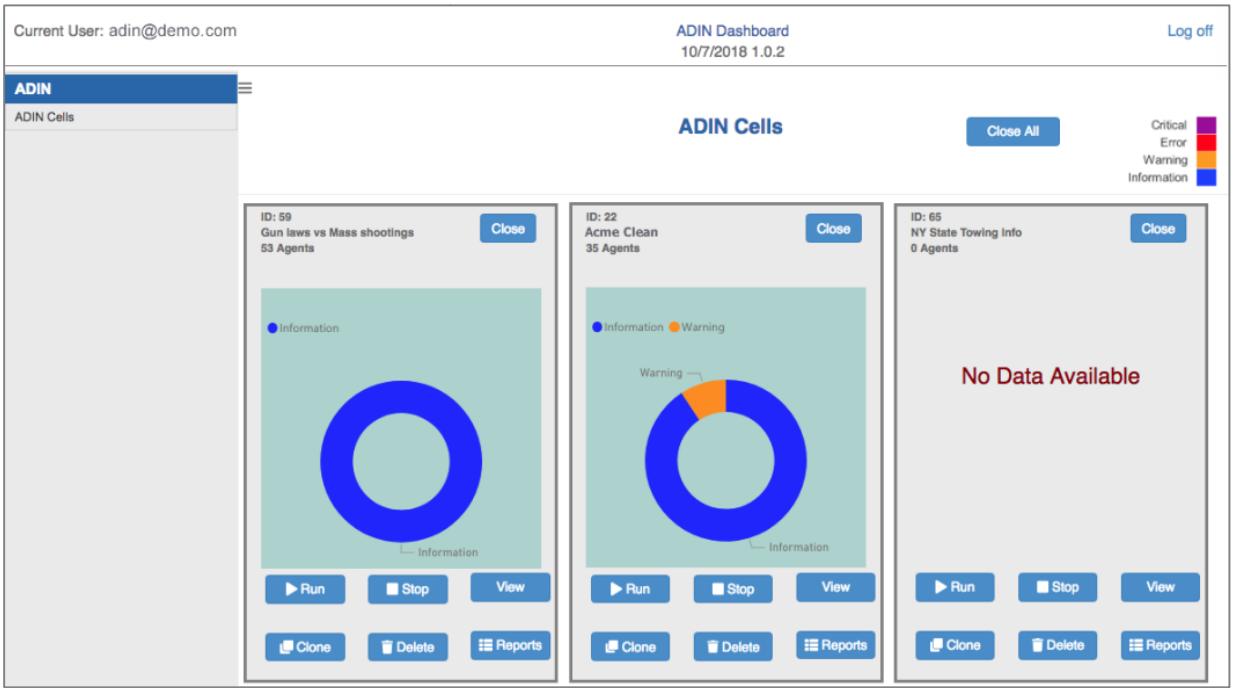

Fig. 3. ADIN is a framework for creating and managing autonomous agents in an intelligent system.

Users define and control ADIN cells composed of autonomous agents reporting on status using color coded dashboards. Each ADIN cell can have any number of autonomous agents defined with any set or combination of triggering criteria based on Data, Time, Location or any combination, as well as multiple user interfaces and data sources (Fig. 4).

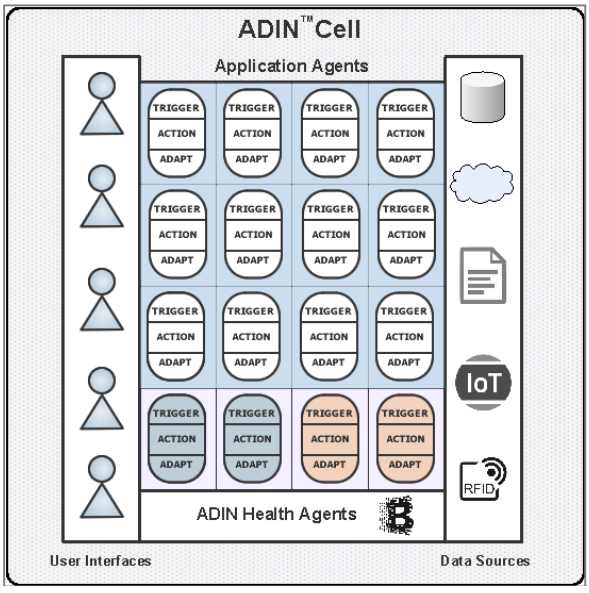

Fig. 4. ADIN cells are organized by common data sources and user interface to solve greater more complex problems involving asynchronous Data, Time and Location

A REST API can be used to automate the creating and management of agents and ADIN cells. The Rest API has 3 main sections: Triggering Criteria, Action Responses and Adaptations as shown in the following Tables 3, 4, and 5.

Table 3. Rest API Library – Agent Triggering Criteria

Agent	Agent Triggering Criteria source is Data, Time, Space, or combo.
Trigger	Data: New Record
Trigger	Data: Updated Record
Trigger	Data: Deleted Record
Trigger	Data: Record fields is a specific value
Trigger	Time: at specific Time of day, Day of Week, Month, Year
Trigger	Time: time before or after another trigger
Trigger	Time: at time interval frequency, i.e. every 1, hour, 6 hours
Trigger	Space: at specific GPS location
Trigger	Space: inside Geo-fence region
Trigger	Space: at location interval frequency, i.e. every 100 feet

Table 4. Rest API Library - Agent Actions

Agent	Agent Actions activate when triggering criteria is met
Action	Create New Record
Action	Delete Record
Action	Update Record
Action	Send notification – template Text, Email, any digital communications
Action	Log results
Action	Log metadata – timestamp, amount and type of data

Table 5. Rest API Library - Agent Adaptations

Agent	Agent Adaptation after actions complete
Adapt	Check triggering criteria more or less frequently
Adapt	Disable agent (self or sibling agent)
Adapt	Activate sibling agent

1.3 Text4Tow Application Example

An application called ‘Text4Tow’ uses an ADIN cell for asynchronous processing to automatically match resources and coordinate the back-and-forth communication when a person is in a disabled vehicle needing roadside assistance. ADIN agents monitor continuously for new service requests using a Text Message based API to request a new tow or roadside service. Table 6 shows examples of the individual agents configured with their triggering criteria and action responses drawn from the agent library from Figure 2.

Table 6. Text4Tow Agent definition

Agent	New Data: Roadside tow requested with GPS location of customer
Trigger	Monitor new records with status 'Unmatched'
Action	Text response to customer to confirm tow request
Action	Find 3 best Matches based on location and available tow operators
Action	Email info to admin for email record
Action	Log metadata – time stamp, amount and type of data

Agent	New Data: Match requested for matched tow operator
Trigger	Monitor match requests with status 'New'
Action	Text/email to matched tow operator requesting tow job with info on customer location
Action	Email info to admin for email record
Action	Log metadata – time stamp, request type, size

Agent	Time after New Match with no accepted operators
Trigger	3 minutes after Match requested without confirmation
Action	Request Next Best matches
Action	Text admin no responses from open requests
Action	Log metadata – time stamp, request type, size

Agent	Status change to Match request – text operator accepted request
Trigger	Monitor match request with status 'Accepted'
Action	Update job status – set to accepted operator
Action	Text to customer, a match is found with estimated arrival based on distance between and traffic conditions
Action	Email info to admin for email record
Action	Log metadata – time stamp, request type, size

Agent	1 mile between tow operator and customer
Trigger	Monitor distance between tow operator and customer location
Action	Text customer tow operator is about to arrive with link to Tow Operator online profile and photo
Action	Log metadata – time stamp, request type, size

The agents in the Text4Tow ADIN cell run unattended and continuously seek each agent's specific triggering criteria and have been in operation for over a year after. The back-and-forth communications between customers and tow operators is 100% automated by agents which greatly reducing back office overhead as well as time. The difference in this time by quickly locating and assigning resources for disabled vehicles can be crucially important to the person stuck on the side of a highway waiting for help.

The resources and time involved creating systems of this level of complexity and asynchronicity is days or weeks to develop, rather than months or even years. Organizing actions around triggering criteria with the built-in ability to adapt means technical human resources focus on actions and their appropriate action responses and less on the infrastructure to manage asynchronous processes. Real world experience shows that patterns quickly emerge in notifications and triggering data are supported by a common agent library that is data aware, event aware, device aware, time aware, API aware, location aware, and more.

Figure 5 shows user interfaces from Text4Tow, a commercial application where asynchronous events are based on a combination of time, location and data events.

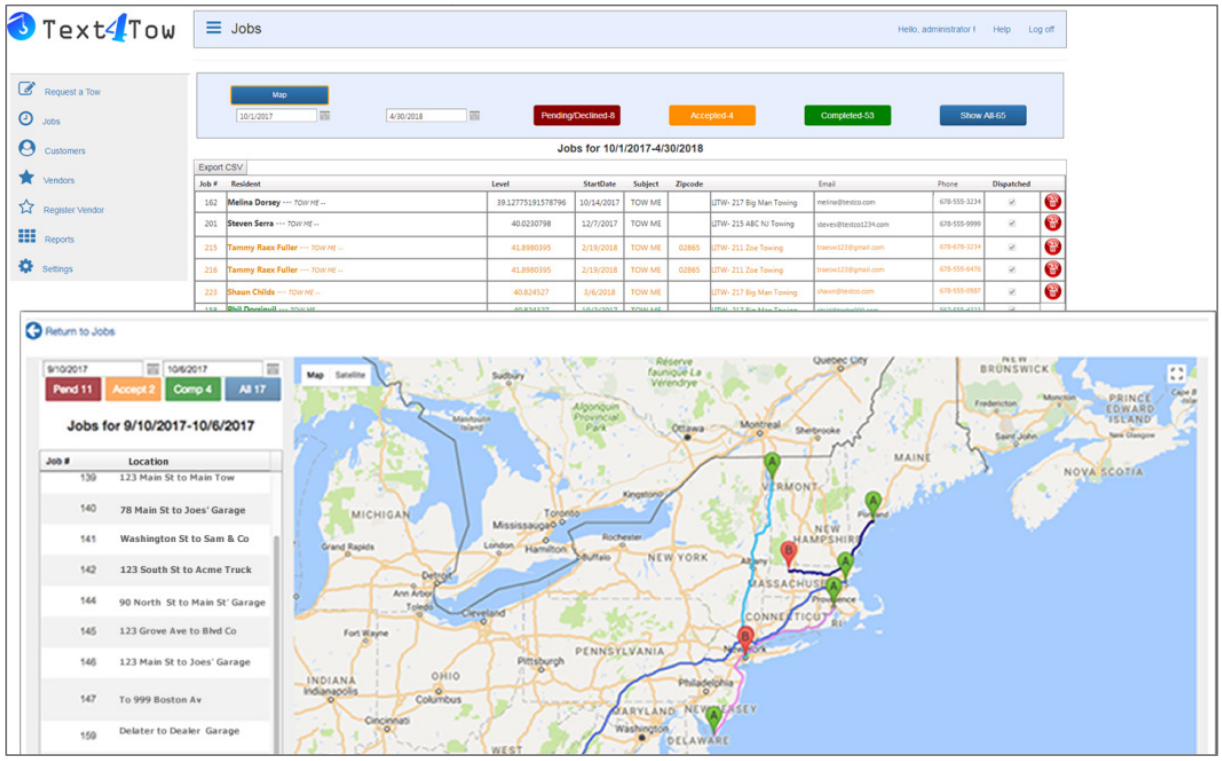

Fig. 5. Text4Tow application based on ADIN agents for location, time, event asynchronous processing

1.4 Virtual Agent Processing Environments

The next evolution of ADIN cells is to scale and expand agents in both complexity and differentiation. With better orchestration of the agent processing environments, application can become much larger and via the Rest API, ADIN cells are created in a more automated way. Individual processing environments for each agent means expanded actions over the agents themselves can be added to the Rest API, such as starting, stopping, pausing and recycling.

Container Technology [2], such as those provided by Docker, Azure, Amazon Web Services (AWS) and others provides an individual virtual processing environment with the necessary amount of computer resources needed by the agent without the overhead of virtual machines. Container orchestration technology such as Kubernetes and Docker Enterprise Edition control specifically where containers are run, such as server, either bare metal or virtual machine.

2 Agent Containers

Agent Containers are ADIN agents [3] running inside a container, of any type as listed in the previous section, where basic ‘proto-agent’ container image run inside containers and defines the basic ADIN agent components of triggering criteria, action response and adaption as shown in Figure 6.

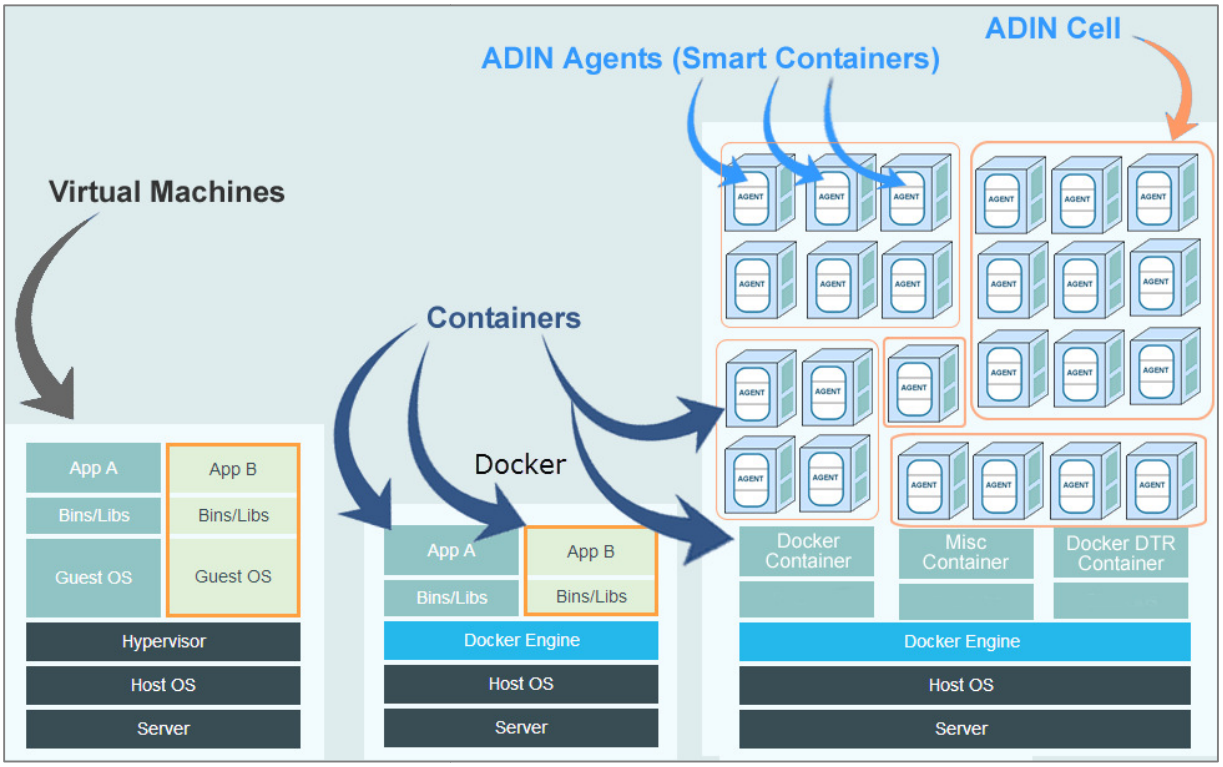

Fig. 6. Containers hold Proto-Agents which are configured with Agent components: Triggering criteria, action response and adaptation.

2.1 Container Technology

Containers are a proven viable solution with fully supported API calls the run ADIN agent proto-agent images. When ADIN cells are now initiated, each agent starts inside its own container and its components are configured into the proto-agent container at startup. Container APIs control starting, stopping, pausing actions on the containers.

ADIN's Rest API was expanded to include these actions are part of the ADIN agent action response.

2.2 Proto-Agents

Agents running in a containerized virtual environments start as Proto-Agents [4] which are agents with blank settings for the triggering criteria, action responses and adaptations. Proto-Agents are configured with instructions read from a datastore or instructed via another process using the Rest API. Agents can direct and control other agents, typically within the same ADIN cell, but for OS level applications, can control agents and ADIN cells in outside of the cell they belong to.

Every agent in this framework has at least one 'Health agent' that tracks normal behavior via metadata and results logging.

2.3 Agent Cloning

Furthermore the ability to clone agents, either individually, in sets or ADIN cell groupings provides important expanded functionality to concepts of intelligent systems and autonomous agents.

Scaling ADIN cells to hold tens of thousands or more agents where the parameter criteria space is automatically covered can be done with Agent Containers and the Rest API configuring all possible set combinations based on the number of parameter dimensions. For example, expanding on the Text4Tow example, say a map of is programmatically segmented into regions based on population and number of tow operators. Adaptations to agents can be configured such that less busy agents are recycled and their territory merged into a neighbor. Likewise, very busy areas can be dynamically refined based on GPS results of disabled vehicles. Agents will respond in real-time to events that happen in nature and are responded to as cars break down.

For example, an ice storm can cause many vehicles to become disabled. Adaptations that are causing high active agents to clone will dynamically increase bandwidth and auto-scale. Automatic cloning with adjustments means Text4Tow auto-responds to the data in the live environment based on events happening outside of the agent's knowledge by responding dynamically to the triggered events.

Recycling and recovery of container agents are also actions that can be part of pre-scripted responses to data and/or processing environments. The built-in nature of containers means that active processes can be paused, set aside and recovered at some point in the future based on the triggering criteria of other agents. This indicates that ADIN cells are well suited to OS level functions and can be the basis for IoT operating systems, which is discussed at the end of this paper.

ADIN's Rest API was expanded to include these actions and is part of the ADIN agent action response as shown in the following Table 7:

Table 7. Rest API Library – Expanded Container Agent Actions

Agent	Agent Actions activate when triggering criteria is met
Action	Start, Stop, Pause, Recycle, Recover Container Agent (self or sibling)
Action	Start, Stop, Pause, Recycle, Recover Container Parent ADIN cell
Action	Clone agent (self or sibling)
Action	Clone and Adjust agent (self or sibling)
Action	Clone parent ADIN cell
Action	Clone and Adjust parent ADIN cell

2.4 Machine Learning and Deep Learning Adaptations

Container Agents created from Proto-Agents are configured via settings for all aspects of their behavior including how they adapt.

Proto-Agents are linked to libraries based on Statistical, Machine Language [5] and Deep Learning [6] algorithms to determine if, and to what extent, adaptations are made as shown in Table 8.

Table 8. Expanded Adaptation Functions

Statistical / Other	Machine Learning (ML)	Deep Learning (DL)
Std Deviation	Linear regression	supervised
Basic Statistical	Logistic regression	semi-supervised
Convolution Filter	Bayesian	unsupervised
Histogram Threshold	Clustering	DNN

Taking the ML expanded adaptations and updating an agent example from the previous section, a new adaption is added that uses ML regression functions over the GPS results from jobs

and will clone itself and adjust setting resulting in two agents now covering the original region as shown in Table 9.

Table 9. Updated Text4Tow Agent Example

Agent	New Data: Roadside tow requested with GPS location of customer
Trigger	Monitor new records with status 'Unmatched'
Action	Text response to customer to confirm tow request
Action	Find 3 best Matches based on location and available tow operators
Action	Email info to admin for email record
Action	Log metadata - time stamp, amount and type of data
Adapt	Function: Linear Regression on logged results of GPS locations; Clone and adjust settings to split region into two regions

ADIN's Rest API was expanded to include ML and DL functions are part of the ADIN agent adaptations as shown in the following Table 10:

Table 10. Rest API Library – Expanded Container Agent Adaptations

Agent	Agent Adaption after actions complete
Adapt	Statistical function, threshold range, response
Adapt	Machine Language function, threshold range, response
Adapt	Deep Learning function, threshold range, response

These adaptation functions are the basis for the Health and Integrity Agents. Logging results and metadata based on normal usage provides the data source over which patterns can be detected for agents seeking to anticipate behavior or to detect anti-patterns or anomalies when things go wrong. This is a key feature of ADIN cells because responding to anomalies in the same time frame as agent processing means agents have a built-in emergency detection system in place. ADIN's Rest API was expanded to include ML and DL functions are part of the ADIN agent adaptations as shown in the following Table 11:

Table 11. Text4Tow Health and Integrity Agent definition

Health Agent	Track normal patterns and trigger on anomalies
Trigger	Calculate statistics over metadata from all agents and monitor for statistical outliers
Action	Text / Email Admin with info
Adapt	ex - avg time to job accept job is 3.5 with Std Deviation = 1.0 Fires If a job acceptance time > avg + 1 STD DEV

Integrity Agent	On startup phase, capture and record signatures and meta data across all agents and store in block chain
Trigger	Calculate and compare signatures and meta data, fire if they dont match with block chain stored versions
Action	Create new Proto-agent, restore original definitions and overwrite agent definitions that have been corrupted

The containerized agents in ADIN cells can be viewed via ADIN Dashboard, as shown in Figure 7, where automatic processing results are color coding to quickly provide the needed insights.

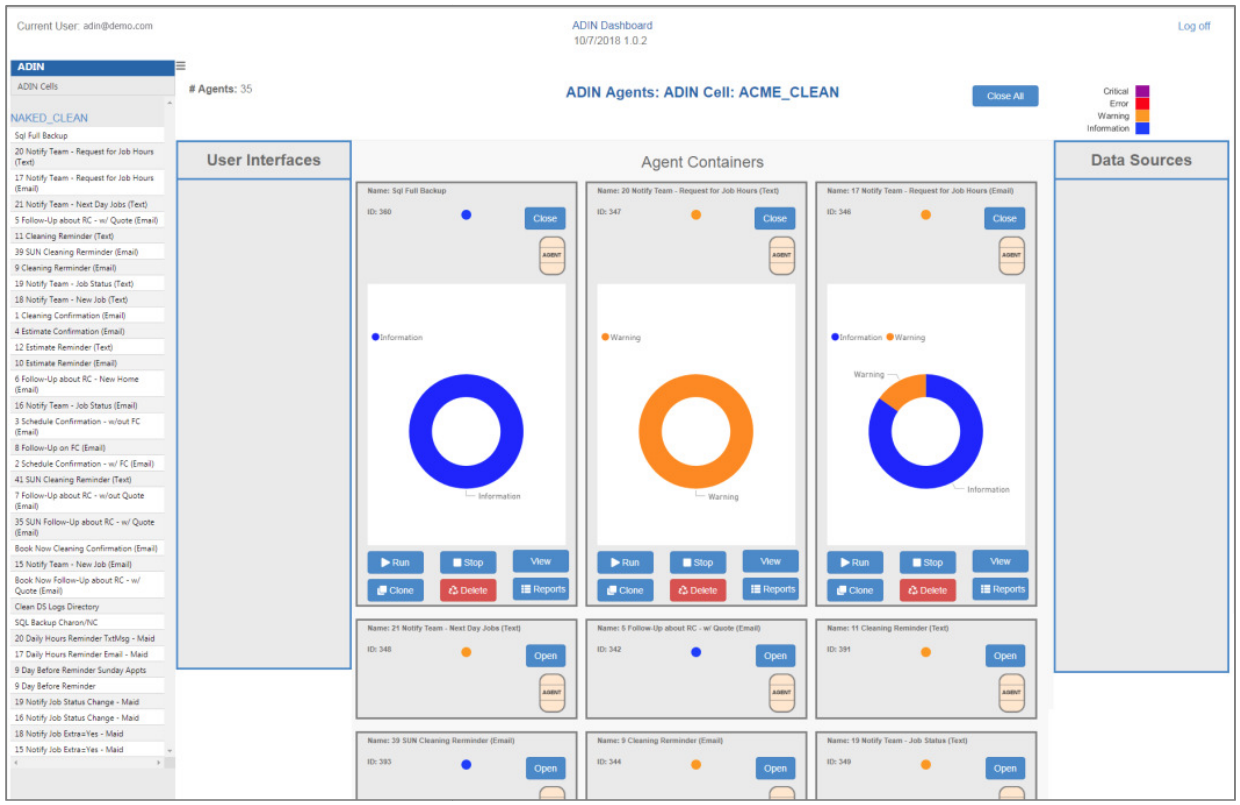

Fig. 7. Dashboard showing agents activity, such as cloning, and cloned-adaptations.

We now have described a framework based intelligent system with autonomous containerized agents with Rest API for programmatic control and Dashboard for insights and manual control. With ML and DL based adaptations built into the container proto-agents, and process orchestration using container technology, we have pushed forward into more complex, dynamic and responsive applications. Two types of applications are particularly well suited to benefit from these expanded capabilities. One is Cognitive applications where automated agents can sift through and seek out structure from the vast unstructured data sources on the internet. The other is IoT applications, where agents can be trained to monitor and analyze any number of devices' behavior and stand ready to respond when anomalies occur.

3 Unstructured to Structured Data

Unstructured data sources, such as those labeled 'Big Data' are digital data sources where structure to extract useful information is not present or insufficient, such as text, logs, web pages, file systems, digital images, digital audio records, databases, etc. Some have built-in structure but don't meet data processing needs. *'Big data are worthless in a vacuum. Its potential value is unlocked only when leveraged to drive decision making.'* [7] For example, keyword-based Internet searches give millions of results sorted by relevance based on some algorithm, but a list of search results is typically a long way from having the needed information presented the most useful way

possible. Furthermore, insightful data can be all but impossible to reach, hidden in relationships among disparate unstructured data sources.

ADIN cells are well suited to applications that need structure from unstructured data sources, especially in cases where parameter dimensionality is high and getting coverage over parameters can be initiated programmatically. For example, an ADIN cells of several hundred containerize agents continuously monitor via online resources US State laws for anything related to gun laws. Also, in the same cell, are agents trained to pull down and add to a database any information related to mass US shootings. There are easily found on the internet and reliably updated by public data sources.

There are standard data triggering criteria based on public web pages and relational, as well as hierarchical. database bases. Mapping tools that are readily accessible will speed the process of locating and mapping data from ad hoc data sources into a standardized schema.

3.1 Cognitive Applications

Below is a diagram of ADIN agents seeking out unstructured data source, extracting results and storing them into mapped structured data source to be used for higher order Cognitive applications,.

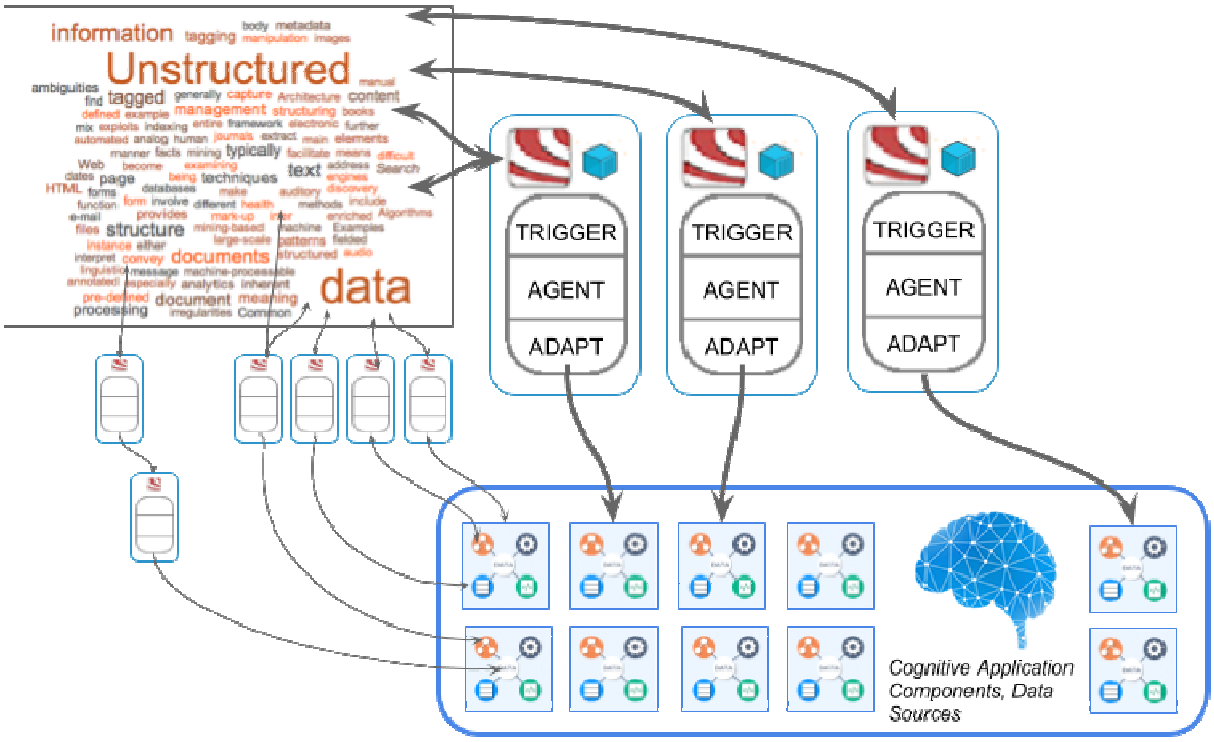

Fig. 8. ADIN Agents creating structured data from various unstructured data sources.

Containerized Agents for unstructured to structured data (see Table 11) monitor various data sources based on their manual or programmed configuration. When new or updated data is detected, the action response is to mine and organize the date into structured data sources that other applications can use. Preprocessing data, such as detecting and removing noise, normalizing for time and/or location, eliminating artifacts introduced as part of the data detection process, all serve to enhance data fidelity and make it ready to be used by higher order processing [8]. Containerized agents adapt to data and processing environment via built-in via container controls and proto-agents

using a Rest API to manage the process of instantiating hundreds, thousands or more agents. Each containerized agent is distinctly configured and reactive, and provides powerful possibilities in today's complex asynchronous application requirements.

Table 11. Unstructured to Structured Data Agent

Agent	Add new / update data from Unstructured data sources to existing Structured data source
Trigger	Connect to database, API, Web services, file system, any digital content
Trigger	Run in real-time, frequently, dynamically, occasionally, on-demand
Action	Synthesize, restructure over sourced data, internal or external
Action	Connect to unstructured or structured data
Action	Notify (email, txt, push, etc) additionally
Adapt	When Amt of data exceeds threshold, clone to increase capacity. When amt drops below threshold, merge / recycle

The resulting structured data sources are ready for to reveal hidden relationships that visualization and interaction via dashboards can prove useful to humans. For example the unstructured data holding gun laws over 50 US states and Google spreadsheets tracking mass shootings, and WHO and US data sources tracking mental health spending per capita by state were used for a dashboard that automatically connected common fields, which in this case where the State name [9]. The resulting interactive dashboard provides a natural way for human to interact by clicking on anything to filter by state, gun law, gender, weapon type, etc. as shown in Figure 9. This is where insights are not always obvious and keeping data fact-based is often difficult. Other areas ADIN cells for unstructured to structured data for visualization has been applied for Resilient Cities, and Climate change.

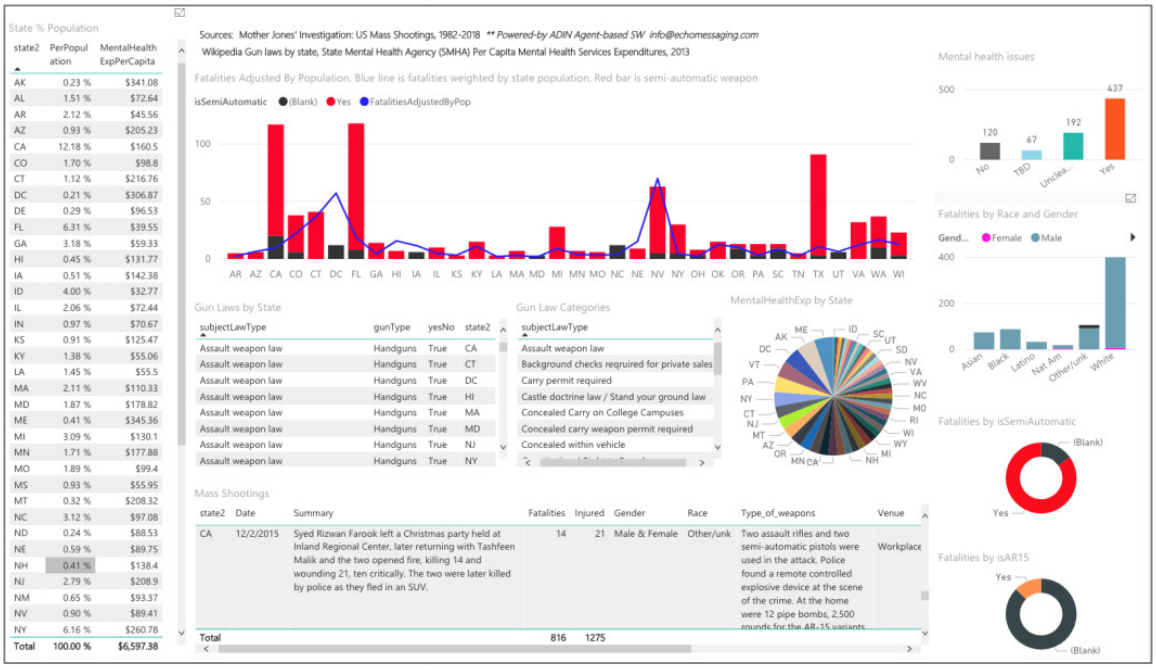

Fig. 9. Interactive Dashboard of structured data resulting from agent processing

Automating the process of creating visual, interactive dashboards by automating the process of creating structure and building relationships from vast disparate data sources leads toward cognitive applications that are the next logical evolutional step in machine learning and ambient intelligence.

3.2 IoT applications

IoT devices are continuing to expand on the Internet due to the number and pervasive nature [10], several problems arise that can benefit from containerize Agents in an intelligent system. These are IoT security via anomaly detection. Agents are programmatically created via the ADIN Rest API starting with one or more proto-agents per IoT device. As new IoT devices are registered or on-boarded, agents can detect the presence of new activity and automatically register devices. A start-up training period based on time-specific agents will monitor for normal behavior. Other agents triggering criteria is to notify on new devices will fire according to their configured behavior. Agents with ML and DL functions seeking to adjust performance based on how IoT devices interact in the real-world will adapt according to the agents configuration.

Since agents' configurations also possible to adjust by other health agents, systems of IoT device management can be done by defining groups of triggering criteria based on data, time, location or any combination. Actions in the event triggering criteria will result in new or updated data, notifications delivered or processing on the agents (either themselves, sibling agents or the parent ADIN cell) are done. ML, DL and statistical processing over logged meta data, logged results are built in to track normal and react to abnormal behavior. An important aspect of this is that the reacting to something wrong happens in the same time frame reference as the process.

When humans are expected to mitigate when something goes wrong, a long time in computer terms can pass allowing for many negative events to occur.

4 Conclusion

4.1 Future Directions

Vast amounts of information are causing problems for consumers of data in that the usability goes down as the amount of data goes up. The Internet is decentralized by nature and has no authoritative body to distinguish the good for the bad, or the useful from the useless. One hopes that the good rises on its own merits (like cream in milk as the old saying goes), but this has not proved true. Deliberate misinformation campaigns push forward corrupt ideas, [11], and the validity of perfectly fine data is susceptible to 'confirmation bias', where people see the conclusions that support their beliefs, while discrediting sources that argue against them [12].

Containerized agents in an intelligent system extract structure from unstructured data sources in an effort to increase their value by distilling down to its core. Cognitive processing automates analytical tasks over vast structured data sources, and presents partial findings and statistically-based outcomes, in an effort to push humanity forward. Just like we no longer care for a horse for transportation, chop wood for heat, or carry water from a well to survive thirst, in the near future, we will not thrive as a technological society if we are so easily influenced by processes over data without understanding why. Ambient Intelligence pushes cognitive-type processing into the barely noticeable background, but when things go wrong, appropriate responses are crucial and required to react at computer speeds.

Presently, leaders and experts are making critical decisions that steer our collective future regarding climate, medicine, global economies, fuel sources, space exploration, and much more, based on big data. Our approach to support cognitive processing is decentralized with built-in health and integrity monitoring. Future directions will continue to promote these efforts.

Devices defining the Internet of Things along with the controls and sensors they entail are adding to the mountain of data at exponential rates. Automated agents controlling and responding

to the important aspects of these devices make them not only useful, but safe. An IoT Operating System based on ADIN cells is underway, where IoT devices have targeted agents that automatically interact, tracking normal behavior, and responding in nano-seconds to abnormal or corrupted behavior.

We have created an AI framework based on containerized autonomous agents in an intelligent system with built-in health and integrity monitoring and have used this framework for numerous, varied commercial systems. Agent libraries provide building blocks for the needs of today and the future to make sense of data by distilling usability based on processing needs. Years of real world applications with common threads of asynchronous events based on time, location, data or any combination that adapt using ML and DL-based algorithms for cognitive processing and IoT applications underpin and validate the design components of the ADIN AI framework system.

References

1. Fuller, T.R., Deane, G.E.: Anomaly detection and intelligent notification. Future of Information and Communications Conference (2018).
2. Dua, R., Raja, A.R. and Kakadia, D.: Virtualization vs containerization to support paas. In: Cloud Engineering (IC2E), 2014 IEEE International Conference on, pp. 610-614. IEEE (2014).
3. Deane, G.E., Fuller T.R.: Multi-agent autonomous system for anomaly detection, intelligent notification and pattern recognition called ADIN. U.S. Provisional Pat. Ser. No filed October 2018.
4. Deane, G.E., Fuller T.R.: Containerized agents in a multi-agent system. U.S. Provisional Pat. Ser. No filed October 2018.
5. Nasrabadi, N.M., Bishop, C.M.: Pattern recognition and machine learning. Journal of Electronic Imaging 16.4 (2007).
6. LeCun, Y., Bengio, Y. and Hinton, G.: Deep learning. nature 521.7553 (2015).
7. Gandomi, A. and Haider, M.: Beyond the hype: Big data concepts, methods, and analytics. International Journal of Information Management 35.2, 137-144 (2015).
8. McCallum, A.: Information extraction: Distilling structured data from unstructured text. Queue 3.9 (2005).
9. Gun Laws Dashboard http://echo.echoware.net/DashGunLaws.html, last accessed 2019/01/20.
10. Internet of Things (IoT) connected devices installed base worldwide from 2015 to 2025 (in billions, https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/, last accessed 2019/01/20.
11. Thai, M.T., Wu, W. and Xiong, H.: Big Data in Complex and Social Networks. CRC Press (2016).
12. Nichols, T.: The death of expertise: The campaign against established knowledge and why it matters. 1st edn. Oxford University Press (2017).