IoT Applications in an Adaptive Intelligent System with Responsive Anomaly Detection

Tammy R. Fuller
Echo Messaging Systems, Inc.
Lincoln, Rhode Island US
tammy@echomessaging.com

Gerald E. Deane
Echo Messaging Systems, Inc.
Lincoln, Rhode Island US
gerald@echomessaging.com

***Abstract*— Components to a robust, secure, reliable and dynamic platform for the Internet of Things includes anticipating for change, and preparing for the unexpected. IoT intends to deliver the next wave of disruption and growth to every facet of digital life, and now is the time for laying the groundwork, upon which IoT processes can be built. In this paper, we present the culmination of over 15 years of Artificial Intelligence research, software development and deployment of commercial applications built using adaptive software agents in an intelligent system framework with automatic anomaly detection and notification. We have created a system of components that have allowed us to create complex applications quickly through the use and re-use of adaptive agents in an intelligent system for automation, communication, and control. To be secure and adaptive, agents were also created whose primary purpose is to monitor normal behavior and react when something out of the ordinary occurs. Reactions can include halting processes, starting up other processes, notifying humans, interacting with 3rd party systems, etc. Each agent includes its own security layer as part of the agent triggering mechanism, as well as the ability to communicate thru all digital channels including Push Notification. Push Notification allows for server-based communication to any number of IoT devices and does not rely upon SIM-based (phone, text messaging) nor IP-based (networking) communication channels.**

***Keywords*— multi-agent intelligent system; autonomous agent; dynamic processing; application framework; artificial intelligence; automated integration; resource allocation, Internet of Things, IoT, IoT middleware platform, push notification**

I. Introduction

Our initial goal was to bring understanding to the Internet - a place where human interaction has exponentially increased over the years with no signs of slowing. Our Intelligent System was rooted in a larger enterprise-level CRM Feedback Management System, called The Echo Messaging System. With components of adaptive behavior and learning, this system has been applied to a larger problem set of, not only communication applications, but also control and automation applications.

Starting with a reusable, adaptive library of plug-in style software components, and a rich intermediate representation, we created a platform of software components and services that has proved to be highly adaptive while lowering the overall time and resources required to build.

With 30 years experience in Artificial Intelligence, we knew it would be worth investing in a reusable, self adjusting structure that would provide returns in overall lower complexity with larger number of potential applications [1].

Our experience showed us that millions of small and medium-sized businesses (SMBs) had expensive yet mundane problems of awkward paper-based manual processes, with data locked into systems that didn't communicate with each other, and an overall lack of streamlined flow of data. We created automation applications that solved the common problems of duplicated entry and basic flow of data, through the continued evolution of the Intelligent System with reusable agents. The more we combined and re-sued the agents, the more companies demanded more complex solutions as shown in Figure 1.

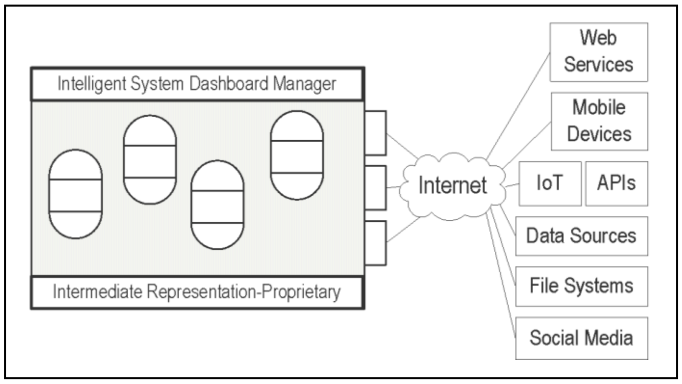


Fig. 1. Agent (re)combine to form applications

The ongoing trend has been for software and hardware manufacturers to include APIs (application programming interfaces) and digital connections to their products and we have been able to expand our library of agents to communicate and control systems and devices via these APIs. The nature of an intelligent system is decentralize the structure to allow for scaling, adaptation, and variation of components to fit the problem set. We started with agents for communication, expanded to newer digital communication channels such as Push Notification, and as APIs became available, expanded into device-control agents. Every agent interacts through a rich intermediate middleware representation, so new agents to control, for example, a Samsung Hub consumer product via their Smart Home Cloud API [2] connects seamlessly to all other agents.

A. *Minimizing Overall Complexity*

By keeping the original agents and continuously adding new ones, we are able to solve larger and larger business problems while keeping the overall complexity of the system minimized.

Like water flowing downhill, the Intelligent System framework with adaptive agents was able to respond to every change brought to us by our customers. Investing in an intelligent system framework, meant that all past efforts were built upon, since only new component agents required new development effort [3].

B. *Software to Think Like People*

At a philosophical level, also rooted in our AI background was always to make the software adapt to the needs of people, rather than force people to think and act like programmers or computers. This meant that agents were created with configurable settings reflecting the needs of the customers. Where customers had similar goals, such as synchronizing financials from an e-commerce into their accounting system, we noticed customers had different ways of interacting with their data. By listening to the needs of our customers, every difference we noted expanded the configuration settings. With new functionality requirements, new agents were created, taking the configuration settings from similar agents as a template. This approach of making the software interact with information the way end users cared about, meant that as the automatic installation of the Intelligent System framework for new applications was already expecting information in a way end users wanted to express them.

In real-world applications, the unexpected must be anticipated. Agents interact with data they were never designed for. Flow of information stops suddenly or peaks without warning. As part of their normal operation, agents in our intelligent system, keep a running history of basic usage, timing and resource statistics. These statistics can be used by other agents to determine 'normal' behavior. For example, an agent, normally sends out 100 emails and text messages for day before reminders for the next day's appointments. 1000 text messages are cued up for delivery. The monitoring agents detect this as an anomaly and it is triggered.. Various responses can occur, such as pausing the delivery of the notifications, sending an emergency notification to a designated human, or any number or combination of responses.

The key concept is to either predetermine the response or pass onto a human to override and/or intervene. Furthermore, the type of human intervention can also be part of a learned response. If the human overrides and allows for 1000 notifications enough times, this is considered 'normal' by the agents. Likewise, if notifications drop off to 0 after normally 100 are delivered every day, a human could be notified. In AI-based systems, it's important to provide the right level of insight into otherwise invisible automatic processes. The feature set that includes the anomaly detection agents has been designated ADAN^{TM} (anomaly detection and notification).

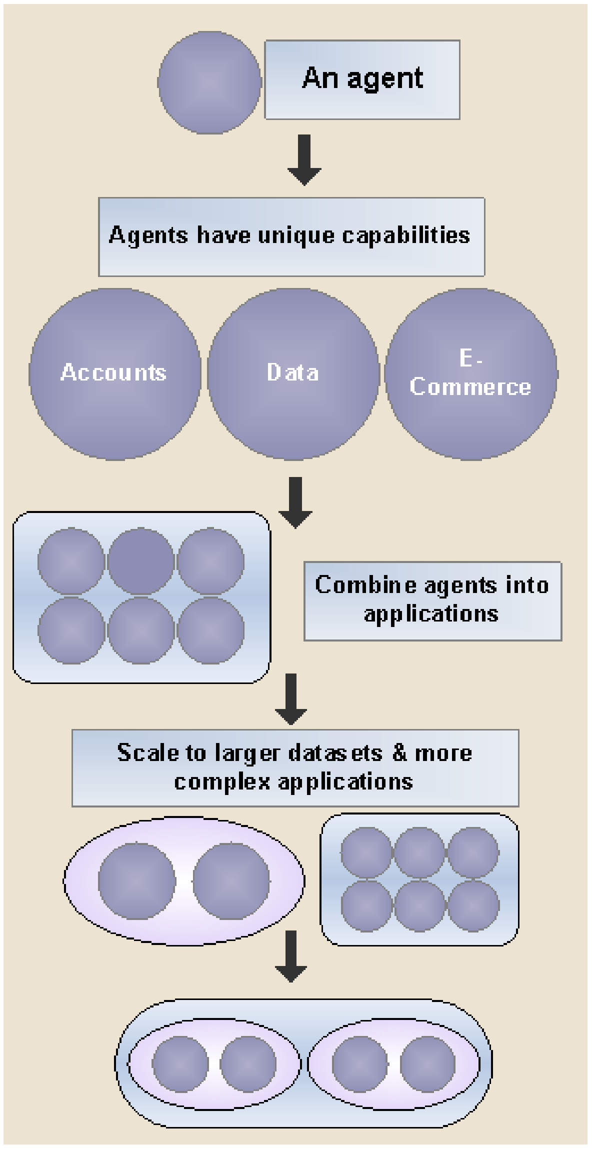

An agent
Agents have unique capabilities
Accounts
Data
E-Commerce
Combine agents into applications
Scale to larger datasets & more complex applications

Fig. 2. Scale to larger more complex applications

II. Adaptive Agents in an Intelligent System

We create applications by defining the application requirements in terms of their goals [4]. Agents from an ever expanding library are added to a new instance of the Intelligent System. Agents are then configured according to the overall application goals and the Intelligent System (IS) starting running. The IS framework is tasked as the environment in which agents run. Agents have the ability to monitor data bandwidth and respond accordingly by making copies of its process, increasing or decreasing sampling times, or deleting themselves when no longer useful.

The IS provides a dashboard manager set of tools to measure IS health and wellness that adequate resources are provided. Some applications run very quietly and perform basic essential background tasks that they become less visible. The IS dashboard provides insight into the actions of the agents if questions arise.

To create software to think like people instead of the other way around, the agents are organized into work units that can be used and reused. The functional 'size' of an agent has been learned over time, in direct response to the needs of customers, and the library of agents divides into the following sections:

- Event aware – agents that monitor for:
    - new records
    - updated records
    - specific statuses
    - combinations of status over one or multiple fields
- API aware – agents that connect to:
    - API, web services
    - RESTFul interfaces
    - Resources where 'software talks to software'.
- Device Aware – agents monitor mobile devices or Internet of Things (IoT) resources.
- Time Aware – agents that track time related to events, such as
    - Daily events – happen 1x per day
    - Continuous events – happen any time of day
    - Countdowns
    - Combinations of time related events.
- Location Aware – agents that monitor location:
    - GPS - Latitude, Longitude, Altitude
    - Address- Address distances
    - Zipcode-Zipcode distances
- Data Aware – agents that connect to data sources such as:
    - Relational databases
    - Hierarchical databases
    - File Systems
- Communication Channel agents for communicating information over channels such as:
    - Email
    - Text Message
    - Phone, Text-to-Speech automated calls
    - Push Notification – direct to mobile devices, bypassing need for SIM card/cell data plans
    - Any digital communication channel
    - Call center integration, for live outbound or inbound calls
    - Social media channels, such as Twitter, Facebook, Instagram, etc

With an expanding agent library, we can create applications quickly by adding agents that fulfill some of the goals. If a new API or type of record is part of the goal, the development time required to create new agents for feature gaps is low as the agents are all template based on the standard set of interfaces.

The goals of an application are realized by an instance of the Intelligent System framework and the configured agents running within the framework. Agents specific to interfaces with the outside world are added as needed, if they are not available in the agent library. Data is interpreted into an intermediate proprietary format, as part of the standard interfaces.

A. Commercial Implications

By combining different .sets of agents into instances of the Intelligent System with various configuration options, we create complex, robust applications that solve the many types of problems common to small and medium-size businesses. Common problems relate to streamlining processes, working with legacy systems and transitioning through the adoption of new technologies. While minimizing time lost due to disruption, which small businesses have very little tolerance for.

As a small business, we found a way to make applications that people need, in a way that is quick, repeatable, robust and reliable, built upon core concept found in AI.

B. Intelligent System Framework

The Intelligent System is a place to house the agents, which do the work. The IS manages the processing of the agents, and provide common resources, and a common internal representation for intermediate data. The IS acts like an operation system of sorts, where agents have a place in which to execute. Depending on the underlying architecture, this could be in a parallel or sequential hardware environment.

Many systems hold the same types of data, even though they perform different types of business logic on that data. For example, all CRM systems will have a first class objects for customers. Different CRM systems will provide specific features related to marketing systems vs. features found in a scheduling system. We created a proprietary intermediate representation that agents interface with. Employing this representation serves to reduce the overall complexity of creating applications.

C. Self-Adapting and Learning Processes

Self adaptation occurs in several places in this system. Agents have an 'adaptation response' phase where they can adjust settings according to their environment. This is encapsulated and the adaptation is specific to the agent functions. For example, a location aware agent could work with an area of focus that could grow or shrink depending on processing results.

Agents are completely encapsulated [5] and it's up to the IS to bring them together to solve the overall application problem or meet the overall goals of the application. Agents have the ability to learn over time if this is part of its overall goal [6]. For example, a location aware agent could monitor daily GPS enabled bus routes to learn their expected arrival times at stops along each respective route. Once the times were learned, the agent would be able to detect an exception to the pattern and issue alert using a communication aware alert such as sending a text message to people who subscribe to the bus route status that it's been delayed.

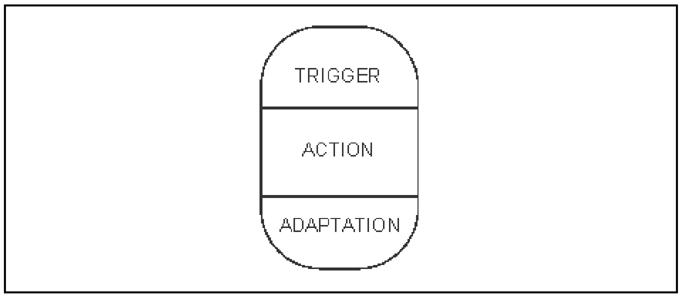


Fig. 3. Agent components

D. Autonomous Agents

Agents' functionality is encapsulated by follow a standard template of three main sections:

1) *Trigger / Goal*
Agents are designed to perform an action in response to a trigger event or goal. The trigger depends on the type of agent and how it's configured. The IS runs the agents according to resources available and gives the agents a processing environment to test their trigger event. If the trigger passes, the associated action is performed next.

2) *Action*
The action can be any type of business logic, connection to any type of API, any type of communication channel connection, etc. Any type of programming activity can become an action. For Data translation agents, the trigger will be configured to determine what type of data is read in and from where (which API, web service, file, device, etc). Data is translated into the proprietary intermediate format. This reduces overall complexity and allows for an architecture of ever expanding library of agents. By having agents translate solely in to and out of the intermediate representation, allows of every new agent to potentially be used to create new applications with older agents.

3) *Adaptation Response*
After the trigger has been detected and the action is completed, there is final phase where the agent can make adjustments to its configuration that will cause it to either self-adapt or will cause other agents to run differently (and hopefully enhanced) when a spike in data occurs, agents can self-replicate or adjust how often they respond to potential triggering events.

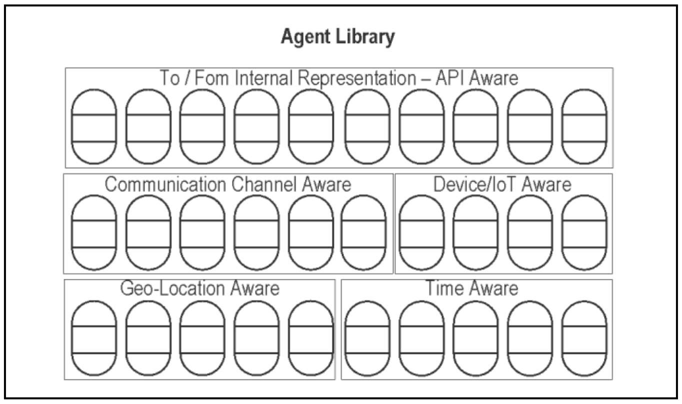
Likewise with cloning themselves, then can end their execution or slow how often they check for triggering events. A goal of the adaptation response is to best use available resources seeking overall system equilibrium and leveled balance of data being processed.

Each agent is configured based the goals they seek to accomplish and the actions they perform. The adaptation responses can change the configurable parameters of its, and/or other agents. The adaptation response can provide logging and health/wellness information to the IS dashboard manager. Adaptation responses can also request further resources.

E. Agent Library

The agent library catalogs agents into goal oriented categories and is constantly growing. The standard format and interaction with the Intermediate Representation means that there is an overall benefit to a large and growing library that allows for the creation of applications based on grouping new combinations of agents.

Fig. 4. Agent library and categories


Agent Library
To / Fom Internal Representation – API Aware
Communication Channel Aware
Device/IoT Aware
Geo-Location Aware
Time Aware

The categories are not limited to what is shown here and new categories are added as needed in response to new application requirements. This approach allows the IS to be dynamic to a continuously evolving IT environment, where agents will be ready for connections to things that don't yet exist, and it is well positioned for working with newer innovations such as 3D printing for additive manufacturing, and new developments related to the Internet of Things.

F. Notification System with PUSH Notification

We have recently expanded into PUSH notification as a new communication channel. Mobile devices (iOS, Android, Windows Mobile, and Blackberry) have an operating system specific Push facility. When a message is 'pushed' is appears in the notification panel and numbered badges optionally appear next to the mobile application.

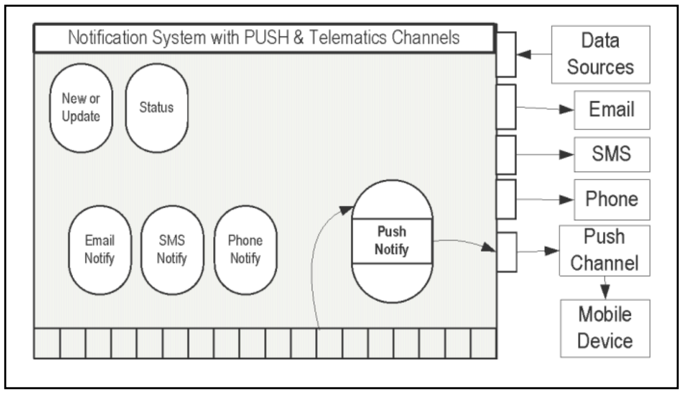
This example shows how we took an existing Notification System and extended key new features by adding it to the existing IS. Push messages are received on mobile devices and the mobile device is first 'registered' to receive push notifications.

This process is transparent to the user. When a mobile application is installed on a mobile device, the Push notification registration is performed and the resulting Push ID which is specific to the device is stored in a central database along with other user information.

The Push Notification agent acts like other notification agents, by triggering from intermediate representation data and initiates the Push message to the recipient. Push messages act similarly to text messages except that a cell phone data plan is not needed to get messages pushed to the mobile device. Also Push messages work with tablets and mobile devices that connect to the internet via Wi-Fi only. This opens up options in terms of the types of applications that are possible.

Fig. 5. DataSync with Notification with PUSH Notification

The Push Notification agent acts like other notification



agents, by triggering from intermediate representation data and initiates the Push message to the recipient. Push messages act similarly to text messages except that a cell phone data plan is not needed to get messages pushed to the mobile device. Also Push messages work with tablets and mobile devices that connect to the internet via Wi-Fi only. This opens up options in terms of the types of applications that are possible.

For example a remote workforce that uses tablets can be pushed messages in real-time without any needs for cell plans.

1) Telematics

Telematics is another area that presents itself as a new digital communication channel. New cars models come equipped with WI-FI, and their center console display panels are essentially mobile tablets, built from the Android, iOS or other mobile platforms. The possible applications are myriad, especially given this approach of rapid application development, with agents already aware of location, time, distance, APIs and various communication channels.

III. IoT Middleware Platform

The EchoWare IoT (Internet of Things) Middleware Platform is based on Echo Messaging System's Autonomous Self-Adapting Agents in an Intelligent System using Artificial Intelligence [7]. The Platform consists of a suite of interoperable software tools whose combined capabilities are well suited for IoT systems. Additional research has resulted in Echo's proprietary ADAN^{TM} - Anomaly Detection and Notification System, which monitors and categorizes normal behavior. Agents that exhibit non-normal behavior will immediately react with preset responses including slow/hard power down, exception reports, device parameter setting adjustment & more.

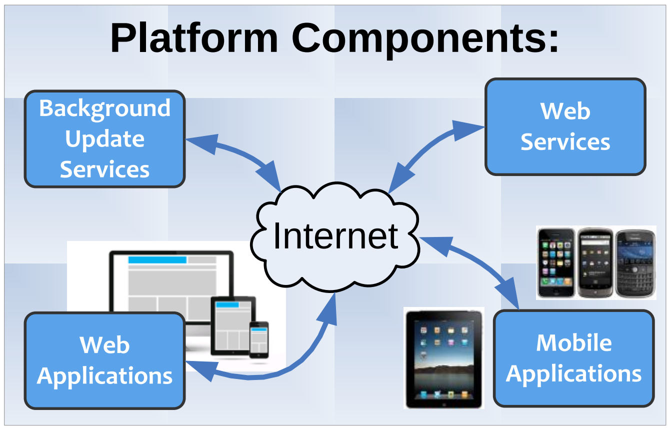


Fig. 6. Agent library and categories

Platform components include web and mobile applications, web services and background automatic update processes.

The Features of this Platform are:

- Plug-in architecture - as discussed in previous sections, the primary benefits to a plug-in architecture are decentralized complexity and scalability
- Alert-based agents - you can have any number of agents, each trained on a specific set of triggering criteria. Each response can be specific to the alert and can be any type of automation, notification and/or control response. Agent react to any type of API event, Timer, GPS location, Data, or more
- Flexible Network topology - the EchoWare (EW) platform allows for IoT devices to be configured in any type of network configuration and connected to any type of 3rd party system, such as:

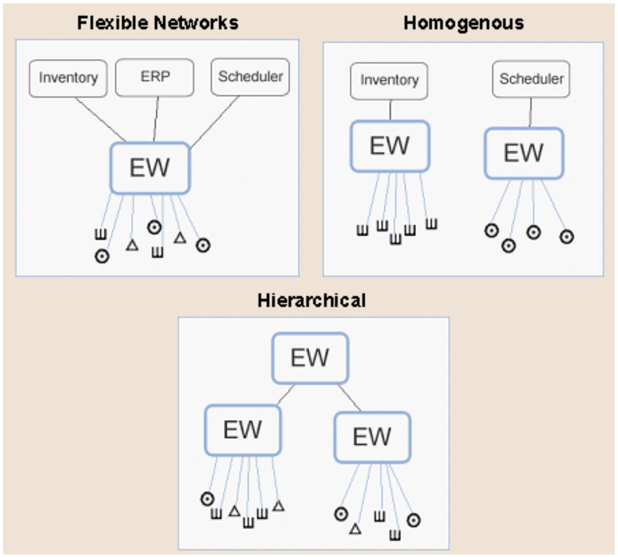


Fig. 7. Various network topologies supported

- Common Command Language - a rich intermediate representation and common set of interfaces allows for agents to be easily combined, re-combined, duplicated with variations, etc.

Communicating in many different formats is a key component of the platform. Formats include:

- Push Notification - push data, media and commands directly to the device from a centralized server. Does not require an IP address, nor SIM card and is therefore unlimited in number of devices to connect to.
- SIM-card based - Phone and text message-based communication, requires SIM card (or equivalent) and a dedicated phone number.
- IP-based - Networking via standard Internet Protocol addresses. Requires network configuration (either dynamic/DHCP or static
- API/Web Services - communicate via APIs, Restful web services and any other software-based API channel.
- New formats - agents can be developed that interface all current capabilities to new formats not yet defined.

Capabilities of this platform available in the various components listed earlier are:

- Register, Self-Register, Batch Register Devices
    - Provision, Security Layer
- Monitor Device & Network health
    - Dashboards, Heartbeat service
- Configure settings
- Exception Reporting
    - Notify and respond to unexpected behavior
- Aggregate Data
    - Common Control Language
- Control Devices:
    - Automatic, Single, Filtered, Batch On-Demand
    - Via Web / Mobile apps, Web services

IV. Push Notification

Push Notification is a set of technologies that allow applications to send notifications directly to mobile devices. This can be used to send messages, media files such as voice and video, commands and application-sensitive events for mobile applications to respond to.

Push Notification uses a mobile device's internal Device ID. Push Notification does not require a SIM card, nor an IP address. This means that non-phone mobile devices can receive information from EchoWare's Agent-based Intelligent system for notifications and commands. Push Notification will be included in the next release of EchoWare.

Devices such as cars, appliances, TVs, kiosks, displays such as those found at sporting events and shopping malls, are a few examples of devices with embedded mobile technology that EchoWare can send Push Notifications to.

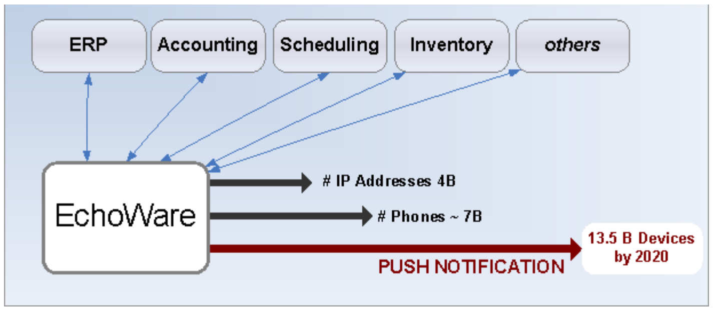


Fig. 8. Push Notification to more devices

1) *Case Study 1- GeoFencing Alert System*

EchoWare has been used to push notifications during an emergency to students and staff with mobile devices registered to receive push notifications. The Push notification included a command to start broadcasting the GPS position 1x/minute. When the mobile device's GPS location crosses a virtual Geo-Fence, the students location and status was updated to school and a push notification can optionally be sent to parents registered devices, which could be their car, indicating that the student is located inside a 'safe' zone.

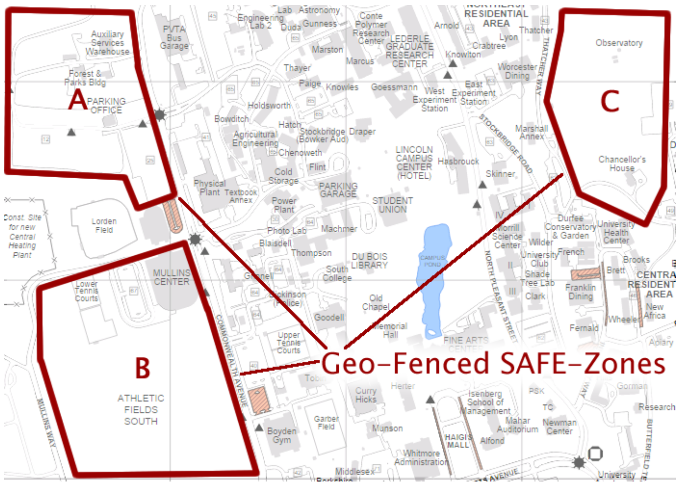


Fig. 9. Geo-Fenced areas designate Safe Zones

Geo-fenced regions are defined as shown in the figure, and are managed by school administrators using a web application.. During emergencies, the school requires everyone on campus to go to a designated safe-zone so people can be accounted for. As part of this system prior to any emergencies, students and staff, download a mobile app onto phone and/or tablets. A SIM-card is not required. Once installed, the mobile device registers with the central server to receive Push Notifications.

The campus-wide alert system communicates to all registered mobile devices that an emergency is underway and to proceed to the nearest 'Safe-zone'. The mobile app then enables a GPS location ping that is monitored by the central server. As devices cross into a safe zone, the users are counted as 'inside the safe-zone' . A web application with mapping feature shows in real-time users inside the safe-zone as well as those outside the safe zone. Once users have crossed into the safe-zone, additional agents are monitoring for triggering condition to send out another push notification to contacts associated with the user to indicate they are now in a safe-zone.

A registered mobile device include cars, as they are now starting to offer internal mobile devices in the dashboard. Cars, like mobile devices are an integral extension of the user and can be used to receive push notifications. This is ideal as mobile devices in cars cannot require a SIM-card, nor an IP-based networking address.

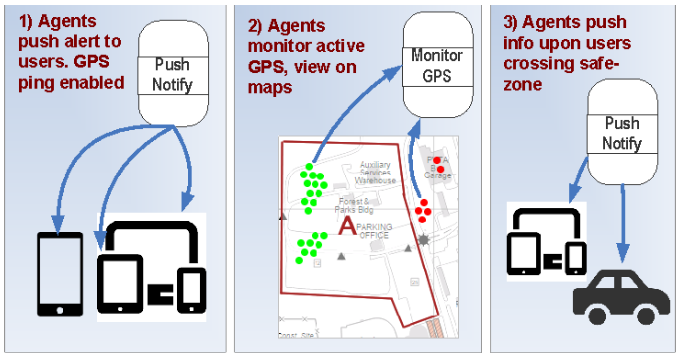

1) Agents push alert to users. GPS ping enabled
Push Notify
2) Agents monitor active GPS, view on maps
Monitor GPS
3) Agents push info upon users crossing safe-zone
Push Notify

Fig. 10. Agents push notifications and monitor GPS

2) Case Study 2 - In/Outdoor GeoFenced Info System

With many overlapping components to Case Study 1, this case study was for an public event that comprised multiple indoor and outdoor venues as part of a larger festival. Each region was defined via a web-base mapping tool. As the registered devices pass into the geo-fenced regions, content, such as text, audio, video, documents or digital media is pushed onto the device describing the highlights of particular region. .

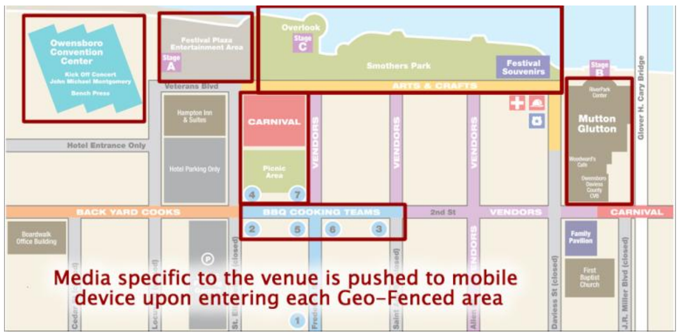

Owensboro Convention Center
Festival Plaza Entertainment Area
Overlook
Smothers Park
Festival Souvenirs
CARNIVAL
Mutton Glutton
BACK YARD COOKS
BBQ COOKING TEAMS
VENDORS
Media specific to the venue is pushed to mobile device upon entering each Geo-Fenced area

Fig. 11. Geo-Fenced areas designate Information Zones

A mobile app receives the pushed content and the user controls if the media is automatically played or on-demand when they desire. Content can be pushed in real-time as new information becomes available, such as weather alerts, or special events.

3) Case Study 3 - Mall Sign Info & Control System

EchoWare has been used to push notifications to pre-registered sets of mall video display signs with personalized ads and messages based on mall-location profile as well as hints based on number and types of viewers. End-users can update profiles to give hints on preferred content to be displayed and frequencies of content updates. Devices are registered to receive Push Notifications using an internal Device ID to respond to direct commands and media content pushed to them. Again, because Push Notifications are used, SIM cards and IP-based networking is not required to communicate with a central server.

Display devices include sensors that can report back status if a crowd size or time viewing at the screen changed, to provide a more dynamic experience for viewers. ADAN^{TM} Anomaly Detection and Notification responds with notifications immediately to administrators, if a display went offline, or if sensors indicated a rapid drop-off or sudden increase in viewers. With agents designed to monitor the health of each device, if a sign indicates a problem or defect, it agent can schedule a service call automatically, as well as notify administrators.

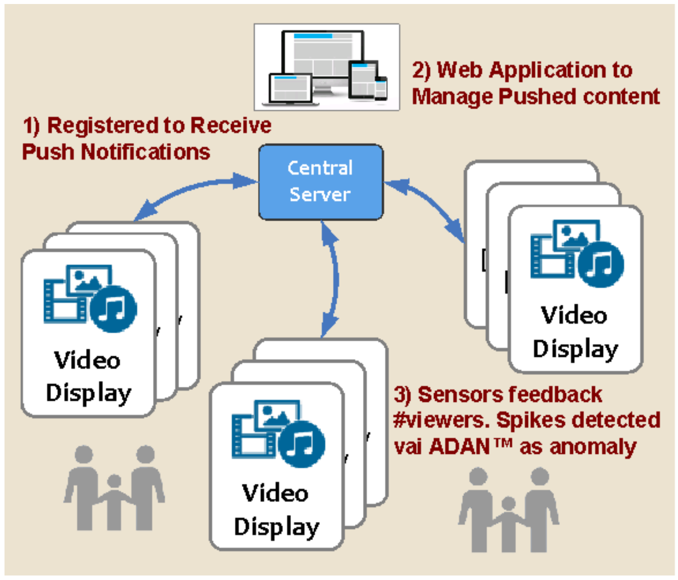

2) Web Application to Manage Pushed content
1) Registered to Receive Push Notifications
Central Server
Video Display
Video Display
Video Display
3) Sensors feedback #viewers. Spikes detected vai ADAN™ as anomaly

Fig. 12. Push content based on #/types of viewers

V. ADAN^{TM} Anomaly Detection and Notification

Autonomous agents in a intelligent system means that most of the work is done quietly in the background. As long as everything is working properly, there is no need for close human intervention. However, when things go wrong, intervention must be quick and appropriate for the condition. If an autonomous agent-based system will affect the well-being of people, or having the system break down means lives are in danger, then it's crucial that these situations be monitored and responded as soon as possible.

AI systems are designed to mirror or mimic human intelligence. EchoWare agents record their basic history for all interaction including how often they are triggered, how long the action phase lasts and what types of adjustments occur over time. ADAN agents monitor this history and activity of agents and learn through a training period, what is consider normal. As EchoWare applications first come online, humans are involved in defining the triggering criteria for agents and determine the normal response. Once agents starting running in their application environment, the local system attributes, sensors and data interaction will indicate what is considered normal behavior. This is one of the ways EchoWare agents mimic human intelligence. Patterns that occur and are followed with regularity give an indicating that all is working normally.

When agent activity ticks up slowly over time, ADAN will not be alerted. However, if a sudden spike or drop-off of activity happens, then ADAN agents are triggered. Some anomalous behavior is normal, such as a notification system that at the end of the month suddenly sends a message to everyone in the system. Administrators will be alerted, but since the system will continue to run, if this pattern is followed at the end of every month, the ADAN agents will pick this up as a normal pattern.

ADAN alerts can be used to monitor the health of devices and to anticipate breakdowns from normal usage. As EchoWare applications scale to larger and more complex applications, having key insight into its operation needs to be integral. Building upon a foundation of pattern matching based on normal usage will provide system administrators both a system wide view, as well as, the ability to drill down into behaviors at the agent-level.

AI systems cause problems when people lose the ability to have insight into automatic processes both from an application breadth-view, as well as the detailed component depth-view. Especially for systems that involve human health and safety, which is happening more and more with the Internet of Things. Having a system at the foundation that is monitoring for anomalous behavior will go a long way to ensure that AI systems serve the greater good for humans, rather than its' demise.

VI. Future Directions

We have focused our attention on creating software systems to meet the needs of a dynamic environment by researching new and better ways to create software, with the goal of solving common problems felt by most small businesses in the everyday world. As the needs of the world grow more demanding, now expanding into the Internet of Things, where people expect more, and demand it more quickly. Companies need new applications faster because technology advances continue to update rapidly, especially with IoT applications. New innovations from the past few years are now commercially available, such as 3D printers for additive manufacture, WI-FI enabled cars with driverless automation options, and IoT applications that connect even more devices to the Internet, are but a few examples of an exciting and dynamic world of innovation. All the while, there are unmet needs in the older technologies used by small and medium sized business that want to benefit from the current climate of innovation.

We plan to make is automate the process of instantiating Intelligent System and selecting the agents based on direct interaction with the end-user. Using voice interactions and guided questions for requirements, the intelligent system could be mostly or completely created. Bringing this entire processing more closely to the end-user will mean more applications are created more quickly. It is possible to envision creating an application in response to end-users verbal requirements, doing the actual work, followed by the entire application going away when done, on an on-demand basis. This may prove to be the best use of resources and merits investigation.

We plan to expand the Agent library to work with technologies that will be in near-term use by a wide variety of businesses such as listed below.

- Internet of Things – agents to:
    - Detect IoT enabled device status changes
    - Newly detected IoT for setup, configuration, registration
    - Control of IoT enabled devices as the triggered action or adaptive response phase
- Additive Manufacturing – agents to work in an advanced manufacturing environment where additive processes print via 3D printers parts on demand.
- Exception Reporting – agents can be created in significant numbers to monitor the activities of individual objects such as planes, trains, etc. and as soon as deviations are detected from expected norms, either predefined or learned over time, can issue exception report notifications immediately. We have started exception reporting in the Find Me Safety™ Platform but we plan to expand further.
- Iterative Adaptive Survey System – The original system from which the IS was created, will be updated to benefit from all the new adaptive features and agents created since 2007. That system is no longer 'too early to market' and will now benefit from an expanded IS framework.
- Parallel Architectures – explore and expand support at the Intelligent System level to harness processing power available in parallel problem architectures.

Self organized applications by describing goals, verbally or through guided questions is another area to consider for future investigation. Once the applications requirements are gathered, the Intelligent System is instantiated and populated with as many agents as possible to meet the goals.

EchoWare will continue to be the basis to solve time, location, and device-based complex applications, through the coordination of adaptive agents via triggering criteria, with specific actions, followed by an adaptation response. From developing and deploying numerous real-world complex applications, we extended a system of monitoring-agents, with the creation of ADAN^{TM} , to promote insight into automatic systems, which we know from direct experience is not only often lacking, but can be very dangerous depending on the application. Expanding into more problem domains, connecting to more devices, and making this technology accessible through more intuitive user interfaces are the next key steps. We are excited to see our vision of seamless and life-enhancing connections between people and things in the digital world, continue to progress and evolve.

REFERENCES

[1] B. Hayes-Roth, 'An architecture for adaptive intelligent systems', Artificial Intelligence, vol. 72, no. 1-2, pp. 329-365, 1995.

[2] "Smart Home Cloud API | SAMSUNG Developers," *Smart Home Cloud API | SAMSUNG Developers*, 2015. [Online]. Available: http://developer.samsung.com/smart-home. [Accessed: 01-Jun-2016].

[3] U. Rutishauser, J. Joller and R. Douglas, 'Control and Learning of Ambience by an Intelligent Building', IEEE Trans. Syst., Man, Cybern. A, vol. 35, no. 1, pp. 121-132, 2005.

[4] H. Zhang, X. Luo, C. Miao, Z. Shen and J. You, 'Adaptive goal selection for agents in dynamic environments', Knowl Inf Syst, vol. 37, no. 3, pp. 665-692, 2013.

[5] A. Ramdane-Cherif, N. Levy and F. Losavio, 'Agent Paradigm for Adaptable Architecture.',JOT, vol. 3, no. 8, p. 169, 2004.

[6] G. Rybina and S. Parondzhanov, 'Modeling of intelligent agent interactions for multiagent systems', Scientific and Technical Information Processing, vol. 37, no. 5, pp. 318-327, 2010.

[7] T. R. Fuller and G. E. Deane, "Creating Complex Applications via Self-Adapting Autonomous Agents in an Intelligent System Framework," *2015 IEEE 9th International Conference on Self-Adaptive and Self-Organizing Systems*, pp. 164–165, Sep. 2015.