NUBILUM AD HOMINEM: Taking smart homes steps further

Chuyuan Zhang*, Yajun Fang†, Berthold K.P. Horn‡
*james.cy.zhang@gmail.com
†Massachusetts Institute of Technology; yajufang@csail.mit.edu
‡Massachusetts Institute of Technology; bkph@csail.mit.edu

***Abstract*—Smart home technologies are developed with the goal of offering convenience to their users, but the currently available technologies are reactive, either to user commands or to user routine, as opposed to being proactive, and are not capable of taking the environment around the user into consideration when making an analysis. In addition, most existing systems have a strictly top-down hierarchical structure, where a central device such as the user's mobile phone controls the system to the point where the lack thereof impacts the functionality of the system on a tremendous scale, and may sometimes result in a cascading failure rendering the entire system inoperable.**

This article focuses on the relationship and connection between various unconnected data points scattered across a person's quotidian life, as well as the decentralized connection between smart home devices, both on a microscopic and on a community scale. It seeks to propose an alternative system that takes into account not only user commands and routines, but also pieces of information disseminated by the surrounding environment, which the user might not have perceived or understood, and incorporates them into its own process of analysis. Further more, the proposed system emphasizes modularization and decentralization, such that the addition and removal of a node should not impact the system outside of the specific functionality provided by the said node.

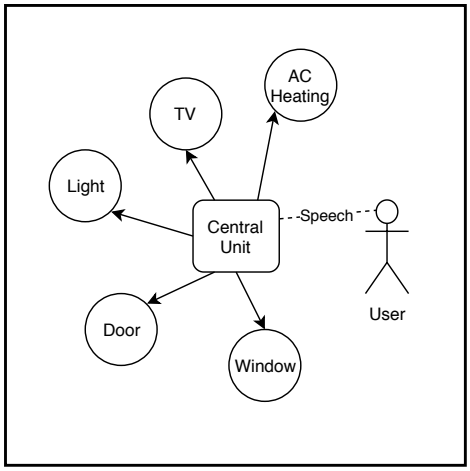

Fig. 1. Smart home system with "level 1" intelligence: the central unit (CU) controls the appliances and relays user command. The arrows represent the unidirectional interaction between the CU and the appliances; the dashed line represents the interaction (command and feedback) between the system (represented by the CU) and the user.

I. INTRODUCTION

The concept of **Internet of Things** (IoT) is a fashion topic in contemporary discourse. Linking physical devices together will make possible further integration of the digital world — the Internet —into our physical world. As an important aspect of IoT, **smart home** has seen the effort of many researchers, and many technologies developed from the research in this field has been made commercially available to consumers.

However, it must be noted that the current smart home technologies are not perfect. The HomeKit™ framework developed by Apple allows the user to control their home appliances from their iOS-powered mobile devices [1]. This paradigm of operation is nonetheless reactive instead of proactive (see Figure 1). User commands are required to trigger the appliances, and therefore the system is incapable of adapting to situations that the user has not foreseen.

Smarter home automation systems exist. Some commercial solutions, such as Nest Thermostats are capable of learning from user routine, and can preemptively adjust itself prior to user action [2]. However, the preemptive behaviour is limited to the appliance; I was unable to find any smart home system that functions preemptively as a unit (see Figure 2).

A major reason for the perceived lack of "intelligence" in existing smart home systems is that they do not utilize the potential of data collection to their fullest. Data collected from mobile payment for a bus fare, for example, can be indicative of whether the user has successfully boarded the bus, and can be utilized in analysis of itineraries.

In order to make the smart home system even more intelligent, it is necessary to make use of the untapped mine that is data collection, and find the relationship between them. Through the linking of various data points, it is possible to create an trigger/event chain that propagates in accomplishing various tasks in a smart home.

In light of this necessity to link various data points scattered across a person's quotidian life together to better serve the user, we propose a system named **NUBILUM AD HOMINEM** (Latin *cloud to person*) — hereinafter abbreviated as **NAH** — which extends upon the existing smart home technologies,

978-1-5386-5197-1/18/$31.00 ©2018 IEEE

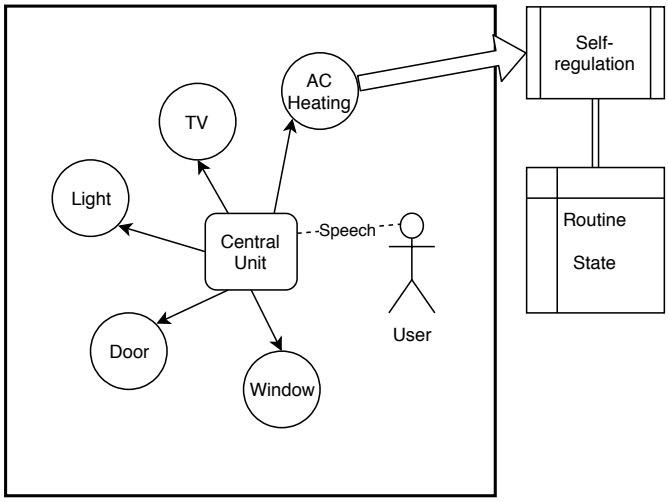

Fig. 2. Smart home system with self-learning and self-regulating appliances ("level 2" intelligence): the appliance can make predictions and adjust itself, but such self-learning does not have an impact on the system as a whole. Similar to that in a "level 1" smart home system, the interaction between the user and the system is done through the CU.

and strives to create a framework and protocol linking not only individual home automation systems, but also the rich resources of the cloud, through connection to external web servers. The interconnected system will enable the connection of disparate data points.

II. Current Smart Home Systems

A. Alexa by Amazon

Amazon Alexa is a virtual assistant developed by Amazon, used in the company's smart speaker product lines, Amazon Echo and the Amazon Echo Dot. Aside from music playback capabilities that comes with the smart speakers, Alexa can analyze and respond to vocal commands that involve pre-installed smart home appliances that are paired with the speaker in advance. The user, through voice interaction with Alexa, can also control some smart-home devices that are designed to be compatible with Amazon Alexa

Strengths: Amazon Alexa sports very strong third-party integration framework through their API, which allows for creation and development of Alexa extensions that the company names "Skills" [3]. Alexa Skills allow for third-party and custom-defined interactions with installed smart-home systems, and allow for extendability beyond functionalities already provided by Amazon. In October 2017, Amazon reported a figure of over 25,000 Skills being developed and published by third-party developers [4]. As of August 27, 2018, over 1000 Skills are developed for various smart home appliances [5], mostly by the manufacturers of those said appliances.

Weaknesses: Given that the smart speaker and the personal assistant form the central focal point of the system, this smart home solution by Amazon is overly centralized. As a result, interaction with the central hub through speech recognition is essential to the functioning of the smart home system. In essence, this restricted behaviour limits the potential user base to only those who speak at least one language supported by Alexa, i.e. speakers of English, German, and Japanese [6]. In addition, the Alexa-centric system is restrictive in its scope; while the smart speaker itself is able to access the Internet, it is unable to apply its Internet connection onto the smart home system. As far as the smart home system is concerned, no outside connection is available yet.

B. Google Home by Google

Google Home is a product line of smart speakers by Google featuring the company's personal assistant, the **Google Assistant**. Similar to its counterpart produced by Amazon, Google Home allows for user interaction with and control of pre-paired smart home devices through voice [7].

Strengths: Google Home is well integrated into other Google services. Its link with Google Calendar assures its competitiveness in handling user calendar events, which is a crucial aspect of users' lives. It is equally capable of being less intrisive and fitting into the home decor. The base of the speaker is a detachable and replaceable fabric shell, and multiple colour variants are sold by Google to be swapped and changed by the user according to the home decor colour theme.

Weaknesses: Many of Google Home's weaknesses are shared with Amazon Alexa. The reliance on user command and thus lack of predictive behaviour is evident on Google Home. Google Home is generally less restrictive in scope than Amazon Alexa, due to the former's ability to perform Google searches, but in a way not dissimilar to that of Alexa, Google Home keeps the Internet functionality separate from its smart home system component.

C. HomeKit by Apple

Apple's **HomeKit** framework lets users configure, control, and interact with their HomeKit-compatible smart-home appliances from their iOS-enabled devices, and as of August 2018, it does just that. The HomeKit framework itself is more of a communication framework than a smart home one, as it offers no intelligent analysis capability itself.

Strengths: HomeKit offers to the user the ability to define room structures and access rights. By defining rooms, the user can more precisely refer to a subgroup of appliances when communicating with Siri, the iOS smart assistant [1], [8]. In addition, the usage of the Home app on an iOS device allows for control outside the domestic wireless network. This makes HomeKit more decentralized than most alternatives, which more often than not revolve around a central hub, usually the smart-speaker [9]. Siri, Apple's speech-recognition-based smart assistant, also incorporates voice fingerprint recognition

that aims to match voice profiles to users, in order to prevent impersonation and access to personal data [10].

Weaknesses: Siri does not have the best speech-recognition quality out of the three commercial products [11], and the HomePod variant is more limited in functionality than the regular version installed on iOS and macOS. In [12], Apple's Siri Speech Recognition Team describes a technology presumably absent on HomePod but present on iOS devices. It is a justified speculation that other features present on iOS Siri may be absent on HomePod.

D. Verdict

As outlined in the sections above, despite the recent progresses made in the smart home system industry, the current commercial solutions share many limitations. The centralized design is prone to fatal failure upon malfunction of the central node. In the case of the three sample systems, if the smart speaker malfunctions, the user will not be able to interact with the appliance verbally. With the exception of Apple HomeKit, which allows for remote control through iOS devices, the user cannot control the system other than through normal methods, which defeats the purpose of the smart home system.

It is worth noting that Amazon, Google, and Apple all design their respective smart system with philosophy of linking it with their other services. However, none of the three companies directly utilizes in their smart-home systems the data generated and gathered in those other services [13], [14]. Apple, for example, despite providing framework and API for third-party developers to utilize its iCloud functions [15], has not integrated this access in their HomePod.

III. NAH Overview

The **NAH** system is designed with the concept of decentralization in mind. Autonomy of individual nodes are preserved to its maximum. The system proposed in [16] offers a basic structure of a smart home system, which can be further extended upon to enable cloud connection and context-awareness. Among other things, it should be possible for devices and appliances to directly communicate among themselves, with the central unit taking on an adjudicator role for conflict resolution.

Definitions

This section introduces several words and phrases mentioned in this article and gives their definition as used here.

- A **domain** is a system of nodes that can function as a single unit with a specific purpose.
- A **node** is a component of a system and a constituent of a domain.
- A **referral** is a token used for authenticating the new node when adding the said node into a domain.
- A **request** and a **command** differ in the assumption made by the issuer. The issuer may not assume completion of a *requested* task, but may assume the *commanded* task to be completed at some point.

NAH Device

An **NAH device** is considered to be an agglomeration of hardware and software components. In reality, the NAH device can take many forms, e.g. thermostats, security cameras, windows, doors. As it is a collection of smaller subdivisions and has a specific purpose, it qualifies the definition of a **domain**. In fact, it is the smallest domain discussed in this article.

The CPU of the device, therefore, is the central unit and commands the rest of the device. This definition makes intuitive sense, as the CPU executes instructions and controls other hardware components. Moreover, we hold the communication with the CPU (through the operating system and other software layers) to be equivalent to communicating with the device itself.

NAH Kernel Modules

To efficiently communicate with the hardware layer in the NAH device, we introduce the **NAH kernel module**s. They are kernel modules that enable direct manipulation of hardware, rather than having to communicate across APIs and ABIs. As they are essentially drivers, they should be loaded as part of the boot sequence.

NAH System

An **NAH system** is a collection of **NAH device**s. Usually a central unit is installed as part of the system to regulate the said system automatically, as well as to transfer user commands and feedbacks. Such a central unit usually takes the form of a home server, which also functions as the central unit for this system (through a specialized module), while retaining its normal function.

Despite the designation as "central", the **central unit** (CU) does not directly control every aspect of the system. It is responsible for processing state changes that cannot be processed by the endpoint which gathered the said data, especially when the state change is not meaningful in itself and needs combined with other data. Additionally, With more computing power than embedded systems in other household appliances, the central home server is capable of performing context-sensitive computations that take into account the surrounding environment from data gathered by the installed sensors..

IV. User interaction

As the ultimate source of authority and the target of the service, the user plays an important role in our NAH system;

thus, interaction between the user and the system is pivotal. To offer maximally efficient communication to the user, the interaction needs to be:

- **bidirectional**: The user and the system need to be able to somehow send messages to each other.
- **spontaneous**: The system needs to anticipate a dialogue at any time.
- **context-aware**: Interpretations of an interaction should also include the context.
- **multi-modal**: Different modes of communication can be included in the same message.

Bidirectional interaction is the easiest to implement; the most basic form of bidirectional interaction is

$$Input \rightarrow Algorithm \rightarrow Output$$

and simply requires the system to give an output to a user input. Through the use of visual and audio cues and responses, the NAH system is capable of conduction bidirectional interactions.

Spontaneity in interaction requires the system to always accept user inputs whenever the user issues them. The usage of interrupts (unless disabled while processing emergency messages) enable this constant vigilance and readiness to engage in interaction with the user. However, to minimize execution time for emergency handlers, user inputs can be disabled, in which case the aforementioned bidirectional interaction becomes unidirectional, with the system notifying the user of the situation and cuing the disabling of user inputs in irrelevant domains.

Context-awareness is what this paper will attempt to enable, through the linking of various data points together. An ideal instance of context-aware interaction would be prompting the user of an alternative itinerary without explicitly asking the user to confirm. By monitoring the user's action, it is possible to determine if the alternative route may be needed, and whether the user would favour the said alternative.

Multi-modal interaction can be established on top of context-awareness, as it requires the construction of data gathered through different methods to be incorporated into one overarching message structure. Patent [17] establishes a way for such interaction to happen on a multi-touch display, meaning that such design philosophy can already be partially realized and implemented.

V. Interoperability

The defining characteristic of a proactive system is the ability to handle events before the user realizes it. The Input/Output Symbolic Transition Systems (IOSTS) outlined in [18] stresses on the modeling of a given reactive system as a whole. An NAH system, however, neither behaves like a monolith nor has a strictly top-down structure; tasks may be transferred among constituent nodes, and the CU steps in only when communication between the nodes cannot be properly resolved. Such a system requires intensive communication, which, as pointed out in [18], requires a standard communication protocol. In NAH, communication is done through **events**.

A. Communication

Two different communication methods should be used. Figure 3 presents an overview of connection types between different nodes, similar to that presented in [16]. As seen in Figure 3, connection between member devices of the same household NAH system can be done through Bluetooth or radio-frequency (RF) links. The advantage lies in the flexible modularization. A node (home device) can be added, removed, or replaced from the system with minimal impact. The central server directly authenticates the addition of a new node and acknowledges the removal of an old node.

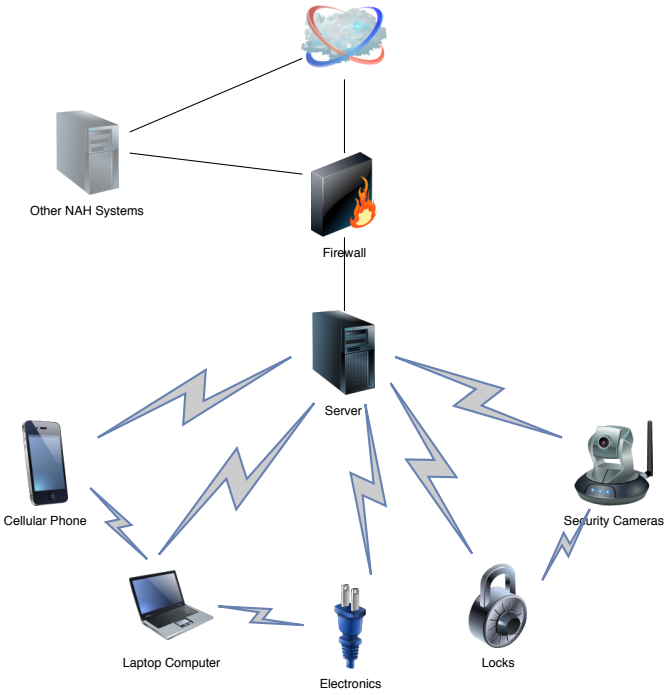

Fig. 3. Network structure

Figure 3 illustrates such connection scheme. For connection to the cloud, Bluetooth and RF are no longer suitable due to the (indeterminate) distance between the nodes. Ethernet as one would normally use to connect to the Internet also should suffice in this scenario. Note the firewall in Figure 3, which is only a preliminary step to ensure safety of the household NAH system. More security measures shall be laid out in Section VII on page 8.

As illustrated in the figure, NAH systems are also capable of communicating directly with each other. This communication can be carried out in radio wave or Bluetooth. Reference [19] describes the ZigBee specification for low-power, short-distance radio communication. Depending on power output

and environmental characteristics, such communication methods can have transmission distances of 10–100 meters line-of-sight [20]. While it is impossible to guarantee line-of-sight between every pair of devices in a realistic home environment, forwarding of messages can circumvent such limitation.

B. Events

Events describe nodes' behaviour to handle a spontaneous state change or to perform a pre-defined task. An event contains one or more triggers and one or more actions. For an event ϵ, we have $\epsilon = (T, A)$ where T represents the set of triggers, and A the set of actions.

Events such as weather change cannot be detected by the sensors installed in a household, but they may be received through weather broadcasts. These events are considered to be "**cloud-triggered**". Some events, such as smoke, should be detected by sensors like smoke detectors. These events are considered to be "**sensor-triggered**". Events are triggered by other events are "**internal-triggered**". These events usually serve in coordinating a multi-node response to an external event.

Abstraction must be made to simplify the representation of an event.

1) Triggers: Triggers ($\tau \in T$), representing a state change, is best described in an **ergative-absolutive**[1] manner where the subject of change is the focal point. They are thus represented by a triplet $\tau = (V, [S], [Arg])$ where

- V (verb) describes the action that triggers the event. e.g. detection of smoke
- S (subject) represents the node or a defined entity that is the subject of the trigger, i.e. the state that has triggered the event
- Arg (argument) describes in detail the trigger.

The $(V, [S], [Arg])$ structure also conforms to the linguistic phenomenon of ergative languages being verb-initial or verb-final [22], as well as the general consensus in the field of linguistics that verb is considered the pivot of a phrase [23], [24].

The structure of O can be user-defined, as long as both the triggering node and the handler understand the same scheme. Generally it is advisable to use a map-like structure, so that individual elements constituting the overall detail can be fetched in isolation.

As an example, the detection of smoke by the smoke detector in the kitchen can be described as $(presence, smoke, (Arg))$, in which Arg might in this case contain supplementary information such as time of detection, air quality, temperature, or video snapshot of the kitchen.

2) Actions: Actions ($\alpha \in A$) are defined much like triggers, but in a **nominative-accusative**[2] manner where the agent of an action (implicitly this node) is the focal point and are represented by the triplet $\alpha = (V, [O], [Arg])$. V and Arg are to be interpreted like their counterparts in triggers. O (object) replaces S to define a possible node or entity to be acted upon (with the agent understood to be the node generating this action).

The similarity lies in the fundamental similarity between triggers and actions. With the exception of sensor-detected state changes and user inputs (which is just a special type of sensor-detected triggers), every definable trigger that initiates an event in one node is, in the triggering node, the action taken as a response to a separate trigger.

3) Semantic interpretation of triggers and actions:

a) 1 element: (V): A trigger taking the form $(V)_\tau$ is interpreted analogously to the sentence "There has been a V-ing" An action taking the form $(V)_\alpha$ is interpreted in English as "I shall V".

b) 2 elements: (V, E): A trigger taking the form $(V, E)_\tau$ is interpreted analogously to the sentence "E has been V-ed". An action taking the form $(V, E)_\alpha$ is interpreted in English as "I shall V E".

c) Special case: Verbs with three slots: Some (transitive) actions have three or more slots (such as in the English sentence "I gave him the book"). They have an agent, a direct object, and an indirect object. In this case, the direct object shall be the second element in the action, and the indirect object shall be contained in the argument.

4) Trigger-Action Equivalence: Since the S field in a trigger and the O field in an action both describe the entity that is being changed, it is possible to draw equivalence between an action and its corresponding trigger. In this section, let $\mathbf{V}$ be the set of all possible verbs in the universe; $\mathbf{V}_t$ be the set of all transitive verbs in the universe; $\mathbf{E}$ be the set of all entities in the universe; $\mathbf{A}$ be the set of all arguments in the universe.

a) 1 element: (v): Establishing the identity $(v)_\alpha = (v)_\tau$ is trivial. The verb v, as the only descriptor of the action, should also describe the trigger, and thus (v) as an action naturally becomes a trigger without change.

b) 2 elements: (v, e): It can be asserted that

$$\forall\, v \in \mathbf{V}_t \,\forall\, e \in \mathbf{E}\ (v, e)_\alpha = (v, e)_\tau$$

because any action done on an entity e (hence e is the object) results in some change in e which acts as the trigger for another event.

c) 3 elements: (v, e, a): It must be conceded that

$$\exists\, a \in \mathbf{A}\ (v, e, a)_\alpha \neq (v, e, a)_\tau$$

[1] A syntactical structure wherein the single argument ("subject") of an intransitive verb behaves like the object of a transitive verb, and differently from the agent of a transitive verb [21].

[2] A syntactical structure where the subject of an intransitive verb behaves grammatically like the agent of a transitive verb but differently from the object of a transitive verb [21].

However, for every argument a, there exists a reciprocal argument a' such that

$$(v, e, a)_\alpha = (v, e, a')_\tau$$

because two-element equivalence has already been established, and the three-element is essentially the same action with more detailed description. Further more, only syntactical arguments may be changed when converting from action to trigger; non-syntactical arguments necessarily describe objective attributes or entities, which must remain invariable.

C. Message Format

As complicated data need to be transmitted between nodes, we encapsulate the messages in a special format containing 7 fields: `header`, `importance`, `expected-completion-time`, `flags`, `timestamp`, `notify`, and `content`.

The `header` field describes the nature of the message. Possible headers include `upd` for updates, `cmd` for commands, `ntf` for notifications, etc.

The `importance` field rates each message in terms of importance. User-defined messages can have `importance` rating of 1-5, with 1 being the most important, and 5 being the least. Messages with `importance` value of 0 also exist, but 0 as a importance value is reserved for emergencies, and — to prevent abuse — user-space applications shall not be able to send or process a importance 0 message without switching to kernel-mode.

The `expected-completion-time` (`ECT`) field contains an integer expressing the expected timestamp by which the sender expects the task to be completed; Since messages with `importance` value of 0 causes the system to immediately process them until completion, the `ECT` values in those messages are discarded. Note that `ECT` is an advisory value, and functions merely as a parameter for scheduling purposes.

`flags` contains various attributes of the message. The lower 4 bits describe the user's role, i.e. how much user interaction is involved in the execution of the task described in the message.

- Bit 0 (`confirm-start`): If set, the user needs to confirm the starting of the task.
- Bit 1 (`confirm-terminate`): If set, the user needs to be confirm the termination of the task.
- Bit 2 (`presence-start`): If set, the user needs to be physically present in the vicinity of the device to start the task.
- Bit 3 (`presence-terminate`): If set, the user needs to be physically present in the vicinity of the device to terminate the task.

The higher 4 bits are as follows:

- Bit 5 (`time-sensitive`): If set, the message is time-sensitive, and needs to be processed before the specified completion time. Priority (see Section VI) is given to messages with this bit set. If the specified deadline is missed, it is generally advisable to drop the task and notify the user.
- Bit 6 ()

The `timestamp` field is an 80-bit value that contains 2 sub-values:

- `seconds`: a 64-bit integer representing the number of seconds since the Unix epoch (1970-01-01 00:00:00)
- `millis`: a 16-bit integer representing the number of milliseconds since the last second mark.

As it takes a minimum of 10 bits to represent every number from 0–999 ($2^9 < 1000 < 2^{10}$), we need a minimum of 2 bytes (=16 bits) for `millis`.

The time represented by the `timestamp` field is precise to the millisecond. Synchronization of time between devices should be done to ensure this precision.

The `notify` field is a 8-bit value, with each bit representing a boolean value. Bits are numbered 0–7 in increased significance. The lower 4 bits specify the audience, and the higher 4 bits define the manner. Note that while this field is considered a *request* rather than a *command* as it cannot be enforced, the recipient is generally expected to oblige, as notification, barring exceptional circumstances, can usually be done in a short time as to avoid disrupting the original task queue.

- Bit 0 (`user`): If set, the message will be forwarded to the user by the recipient.
- Bit 1 (`broadcast`): If set, the message will be sent to every node connected to the NAH system.
- Bit 2 (`sender`): If set, a copy of the message is returned to the sender.
- Bit 3: Always 0.
- Bit 4 (`verbatim`): If set, the recipient is asked not to change the message; otherwise the recipient is permitted to update the timestamp before forwarding the message. In both cases the recipient may clear the `notify` flags.
- Bit 5 (`strong-verbatim`): If set, the recipient is asked not change the message *at all*; each field, including the `notify` field, should be forwarded as-is. This bit is to be ignored if the `verbatim` bit is not set.
- Bit 6 (`time-sensitive`): If set, the message is considered time-sensitive and need not be cached; otherwise, the message needs to be temporarily cached by the recipient if it cannot be forwarded immediately.
- Bit 7 (`forward-first`): If set, the message needs to be forwarded first before the recipient can process the contents.

The `content` field contains actual content of the message. Since this field includes information specific for each application, no standard to regulate the use of this field has been prescribed, and it is intentionally left fully customizable, allowing different applications to define unique formats.

Universally Unique Identifiers (UUIDs) are used to identify the message, the sender, and the recipient. The assignment

of every NAH device (node) with a UUID ensures that identification of such a device is context-insensitive (i.e. the ID does not change based on the context). It also allows the nodes to be treated just like another entity when being processed, and enables reference to nodes and other identities in event (see Section V-B).

The fields `sender-uuid` and `recipient-id` might not designate the sender and recipient of a transmission, especially in the case of forwarded messages. Nodes handling a message must always refer to the UUID to identify the interested parties. Exceptions are made for encryption/decryption/signature purposes. The node should decrypt the message using credentials of the sending node, upon which the actual sender (creator of the message) is revealed.

D. Event chains

The processing of an event in its entirety cannot be done on one node alone, and as such, the initial node must delegate some of its work to other nodes. It is thus important to define a behavioural protocol for each defined event, which the nodes shall adhere to. Such a protocol takes the form of an event chain linked by other triggers.

A household NAH system, tasked with home maintenance, is perceptive to triggers from multiple areas including weather, security, navigation, and network. **Initial triggers**, i.e. triggers that set off the event chain, are usually predefined, but messages to the receiving NAH system can also contain ad-hoc instruction of what is expected to be done. The downside of these ad-hoc instructions is that it requires the sender to have detailed knowledge of what the receiving system is capable of, lest the instruction be unrealistic for the recipient to execute.

Since the propagation of such an event chain is done through the use of messages, from the perspective of a node, the initial trigger is the reception of a message. The message is first parsed one level down, fetching the metadata and storing the contents. The message is then inserted into a queue according to the priority value associated with it. The actual parsing of the message contents (i.e. the `content` field) is not done until right before actual processing of the said message. The `content` field is parsed with a custom-defined parser, as the field itself does not necessarity follow JavaScript Object Notation (JSON) standard.

As shown in Figure 4, the event chains are not unidirectional, but rather accommodates feedbacks and overrides. The user is able to override the decision made by the NAH system at any time, and the

As mentioned in Section III, system can behave like nodes on a larger scale. In a system, there is usually at least one node that is capable of receiving messages from beyond the system boundary. In more centralized variants such as a household NAH system, the central unit fulfills this role, and essentially acts as the spokesperson of the system. In distributed variants, such as a community network, every node

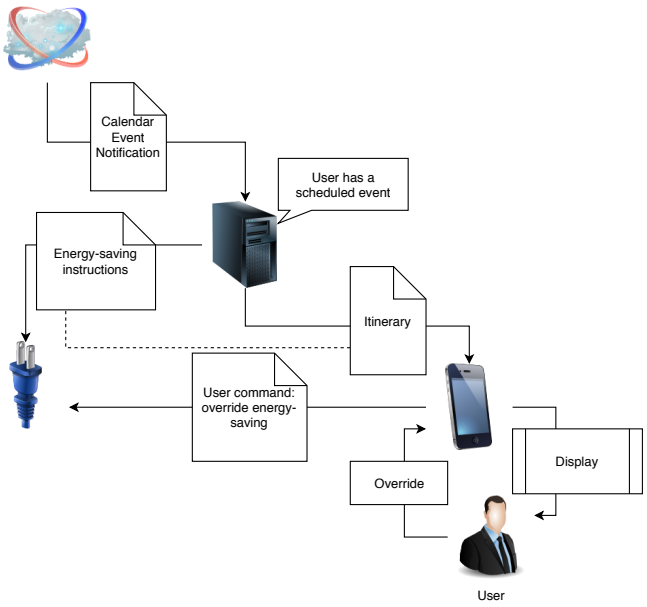

Fig. 4. Sample event chain

is capable of receiving information and messages from outside the network. From the perspective of an outside observer, the system behaves in an analogous way as a node: message is received; processing is done; feedback is issued.

VI. Scheduling

Every stored message at any given time can have an internal priority value based on the current state S and the message M, of which 3 parameters (`importance`, `urgency`, `user-role`) influence the scheduling decision. For such a message, we can assign it a priority value $p = Priority(S, M)$ where $Priority(S, M)$ is a function.

The state S may include current timestamp, household activity information, weather, energy consumption, etc., while the message can essentially be abstracted as its metadata information.

Different scheduling disciplines exist. The following subsections explores the major ones and analyzes their strengths and weaknesses in an NAH paradigm, before offering a solution.

A. Scheduling Disciplines

1) Round-Robin: **Round-robin** (RR) is a scheduler in which time quanta are assigned to the process equally and in a circular order [25, pp. 69–70]. It is considered a preemptive algorithm the process is preempted, i.e. interrupted without its cooperation, once the time quantum assigned to the process expires; and indiscriminate in that it favours no particular process. In an NAH system, while tasks with same importance generally should be favoured equally, importance does exist

as a parameter, which makes RR somewhat limited as a scheduler.

2) Earliest Deadline First: **Earliest deadline first** (EDF) is a scheduler that, upon completion of an executing task or insertion of a new task, searches through the process list and schedules to be next be executed the process closest to its deadline. While EDF minimizes the occurrence of missed deadlines, in our NAH system, "deadlines" are more advisory than compulsory. EDF's consideration of only one metric (deadline) is thus a noticeable weakness in this case, making a pure EDF scheduler unsuitable for the purpose of NAH systems.

3) Multilevel Queue: **Multilevel queue** is a queue with a predefined number of levels where processes are assigned to a particular level at insert by a level assignment algorithm. It is flexible in that each level (which is itself a queue) can have a different scheduling algorithm. e.g. the lowest level can use RR while still respecting the necessity for priority levels.

4) Multilevel Feedback Queue: **Multilevel feedback queuing** places processes dynamically in a set of queues and places a process in a lower queue whenever it uses up the time quantum [26]. It was designed to favour short processes. However, MFQ treats all processes equally when they are entered into the system: every process is first pushed into the highest queue. In an NAH system, importance and urgency differences exist, making the MFQ as-is unsuitable.

B. NAH Scheduler

As **multilevel queue** allows inclusion of other algorithms for queue-specific scheduler, permitting higher flexibility, the overarching model of an NAH scheduler should be a multilevel queue. Given that there are 5 importance values (0, being emergency, signals an immediate execution), there should be at least 5 queues in the scheduler.

VII. Transmission Security

In the process of linking various data points of the user's life together, it is necessary to connect the NAH systems to the Internet. As such, transmission of private data is often inevitable. Protecting the user's privacy in this case is equivalent to ensuring the security of user data. Such security measures come in two parts: security in the process of transmission, and confidence in the nodes participating in such communication. Considering that the Internet as a whole is unsecured and open, measures must be taken to ensure both the security and the integrity of transmitted data.

In **asymmetric cryptographic systems**, the public key is distributed while the private key never leaves the host. Since only the private key can be used to decrypt the message, interception of the public key does not enable the interceptor to know the actual contents of the message. Asymmetric cryptography, however, relies on trapdoor functions, and as such, the generation of a public-private key pair can be computationally expensive.

Symmetric cryptographic systems do not rely on the existence of trapdoor functions. Such independence results in faster key generation, but since the same key is used to both encrypt and decrypt the message, transmission of the said key is susceptible to interception by a malicious third party, an issue not present in asymmetric cryptography.

It is thus advisable first to use an asymmetric cryptographic algorithm to distribute a key that is shared by everyone in the dialogue, and then use a symmetric algorithm to perform actual encryption and decryption. A method known as Diffie–Hellman key exchange (DH) can be utilized for this purpose. Compared to RSA, DH is faster, but it is not an actual cryptographic system, but rather a procedure of securely negotiating to create a key.

A. RSA

RSA (**Rivest–Shamir–Adleman**) is an asymmetric cryptosystem and is regarded as an iconic example of its kind. RSA's asymmetry is based on the "factoring problem": the factorization of the product of two large prime numbers. With correct implementation (and a large enough key), decryption of a message that has been encrypted with the public key cannot be accomplished without knowledge of the corresponding private key [27].

Attacks against RSA: Attacks against RSA has been made in the following ways:

- If the original message m and the encrypting exponent e are small ($m < n^{\frac{1}{e}}$) to the point where m^e is less than the modulus n, then decryption becomes easy by taking the eth root of the ciphertext c because $c = m^e$
- RSA's deterministic nature makes it prone to chosen plaintext attacks. Without secure, randomized padding, the attacker can test likely plaintexts and compare them against the ciphertext [28].
- Another approach is by exploiting the identity that ${m_1}^e {m_2}^e \equiv (m_1 m_2)^e \pmod n$. To learn the decryption of ciphertext $c \equiv m^e \pmod n$, the attacker may choose a value r and ask the decrypting party to decrypt $c' \equiv cr^e \pmod n$. Since we have

  $$c' \equiv cr^e \equiv m^e r^e \equiv (mr)^e \pmod n$$

  the decryption of c' is $mr \pmod n$ from which the attacker can deduce the value of m.

B. Diffie–Hellman Key Exchange

The original implementation of Diffie–Hellman [29] relies on the idea of multiplicative groups of integers modulo n, and assumes that only two parties are communicating.

Parties 1 and 2 first agree on a modulus p and a base g such that g is a primitive root modulo p.

Party 1 chooses a secret integer a and computes value of $A = g^a \bmod p$. Party 1 then sends A to Party 2.

Party 2 chooses a secret integer b and computes value of $B = g^b \bmod p$. Party 2 then sends B to Party 1.

The final key can be computed by the two parties separately by computing the received value to the power of their respective secret integers, and finding the remainder modulo p. i.e. Party 1 computes $A' = B^a \bmod p$ while Party 2 computes $B' = A^b \bmod p$.

This method used to be considered secure because while the values of p, g, A, and B are transmitted and thus assumed to have been compromised, as computation of the final key requires knowledge of at least 1 private value (which should never be transmitted), it was considered virtually impossible[3] for a third party to know the value of the key.

D. Adrian, *et al.*, in [30], assert that with enough pre-computation, brute-force attacks on Diffie–Hellman is feasible. The authors estimate such pre-computation for decrypting Internet traffic to cost US$100 million, making it a feasible option for large intelligence agencies such as the NSA (hence not "NSA-proof"). Notwithstanding the estimated cost, if we limit such exchange of keys to the initial pairing stage, we can opt to use a more complicated base without impacting the efficiency of the system. While it is computationally expensive to generate large prime numbers dynamically, there are ways to expedite such generation. Java 8 big integer library, for example, is able to generate a large (2048-bit) prime for use in DH exchange in approximately 3 seconds.

C. Advanced Encryption Standard

The Advanced Encryption Standard (AES) is a block-cipher specification for cryptographic systems. It is based on Rijndael, which is a block-cipher encryption algorithm[31]. While many use the term AES and the original name of the algorithm, Rijndael, interchangeably, the AES only contains 3 valid key lengths (128, 192, and 256 bits) and 1 valid block size (128 bits), whereas Riindael is can have any block size that is a multiple of 32 bits.

Rijndael (and therefore AES) is based on the principle of substitution-permutation network (SP-network)[32], where multiple rounds, each containing substitution and permutation operations, is applied onto the input (plaintext and key). In AES, the SP-network is manifested in the form of matrix operations on a 4×4 matrix called a **state**.

The number of rounds is also defined in the AES with respect to key length:

- 10 rounds for 128-bit keys.
- 12 rounds for 192-bit keys.
- 14 rounds for 256-bit keys.

Reference [31] contains in detail the implementation for AES. In general, the S-Box and its inverse are stored in memory as lookup tables, instead of computed.

[3]With the exception of brute-force, which is described in the following paragraph

Attacks on AES: Software-level AES implementations are prone to side-channels attacks, which are not considered in classic cryptography settings, but rather utilize implementation-specific data (such as CPU temperature, reaction time). So far, side-channel attacks such as that by G. Irazoqui and X. Guo, presented in [33], have been the only successful attacks on full AES-compliant algorithms. According to [34] published in 2006, the best known cryptographic attacks were on 7 rounds for 128-bit keys, 8 rounds for 192-bit keys, and 9 rounds for 256-bit keys. It is however possible (and already done by Intel, as published in their white-paper [35]) to implement hardware-level AES instructions which can limit the feasibility of side-channel attacks.

VIII. Authentication

While the security measures in transmission has been described in Section VII, the more urgent concern is the inadvertent transmission of private data, which includes communication with an impersonated client. In this case, authentication using asymmetric key pairs is important. However, confidence in the communication sense includes not only the integrity of the message — which is verified by employing signatures — but also the confidence in a node's identity, i.e. the process of authentication.

A. OpenID

OpenID is a standard and protocol for decentralized authentication that enables users to be authenticated through a third-party server known as the identity provider [36].

If a user U wishes to log into website W, instead of directly logging in, W redirects U to a webpage maintained by an identity provider I, where U enters his/her credential for I (analogous to a normal login process into I). I then confirms U's identity to W. Upon such confirmation, W grants U access.

Flaws: The most obvious attack against OpenID is phishing [37], a process in which the attacker A uses a fake login page — pretending to be the identity provider I — to trick the user U into entering the credentials. With knowledge of such credentials, A is then able to impersonate U when communicating with W.

Also, the redirect URL from the identity provider to the service provider can be replayed to allow a third party wire-sniffer to log in as the user, if the connection is not secured with TLS/SSL. Even with the usage of nonces, active attackers can still impersonate the user by logging into the website before the user does [38].

B. Signatures

Authentication through OpenID is usually done when initiating a conversation; it is infeasible to authenticate every single message this way. By signing the message (attaching a signature to it), the sender is able to prove its identity.

Signatures use aspects of asymmetric cryptography; in fact, the most widespread way of signature is the reverse of RSA. The sender may produce a hash value h of the message m and encrypt it with the private key. The recipient decrypts the signature with the sender's public key and compares the resulting value h_{sig} with the actual hash value of the message h_m. If $h_{sig} = h_m$ then the recipient can be sure that the sender possesses the private key of the intended sender, and that the message has not be tempered with.

IX. Conclusion and Future Work

Certainly, this article presents merely a vision, and many details are missing from this article. As of now, the following work still needs to be done:

In order for the smart home system to be able to trigger event chains on its own (to perform anticipative actions), it needs to be able to detect and analyze the states on its own. Data like temperature and humidity are easy to gather with a specialized sensor, but events like user activity must be inferred from crude data. Such inference, known as activity detection, should be a major component in the functioning of the NAH system.

It is also important to evaluate possible ways of conserving the power. The introduction of NAH into people's lives will increase household power consumption. Minimizing such increase is important to both the introduction of NAH system and the environment.

Lastly, compatibility with current systems must be considered. The structure and vision in this article assumes communication with other NAH-compatible systems. In reality it is impractical to assume that all systems will be NAH-compatible.

References

[1] R. Ritchie, "HomeKit in iOS 8: Explained," *iMore*, Aug. 27, 2014. [Online]. Available: https://www.imore.com/homekit-ios-8-explained.

[2] D. Pogue, "A Thermostat That's Clever, Not Clunky," *The New York Times*, Nov. 30, 2011.

[3] Amazon. (2018). Understand the Smart Home Skill API, [Online]. Available: https://developer.amazon.com/docs/smarthome/understand-the-smart-home-skill-api.html.

[4] Business Wire, "Amazon.com Announces Third Quarter Sales up 34% to $43.7 Billion," *Business Wire*, Oct. 26, 2017. [Online]. Available: https://www.businesswire.com/news/home/20171026006422/en/Amazon.com-Announces-Quarter-Sales-34-43.7-Billion.

[5] Amazon. (2018). Smart Home: Alexa Skills, [Online]. Available: https://www.amazon.com/Alexa-Skills-Smart-Home/b?ie=UTF8&node=14284863011.

[6] A. Hartmans, "Amazon wants your help teaching Alexa new languages — and it could help in its fight against Google," *Business Insider*, Mar. 20, 2018. [Online]. Available: https://www.businessinsider.com/cleo-new-alexa-skill-for-teaching-foreign-languages-2018-3.

[7] D. Bohn, "Google Home: a speaker to finally take on the Amazon Echo," *The Verge*, May 18, 2017. [Online]. Available: https://www.theverge.com/2016/5/18/11688376/google-home-speaker-announced-virtual-assistant-io-2016.

[8] M. Wollerton, "Control your house with Apple's Siri-enabled software platform," *CNET*, Sep. 10, 2015. [Online]. Available: https://www.cnet.com/reviews/apple-homekit-preview/.

[9] R. Crist, "What a streamlined HomeKit means for Apple's smart home ambitions," *CNET*, Jul. 21, 2018. [Online]. Available: https://www.cnet.com/news/apple-homekit-software-authentication-explained/.

[10] Siri Team, "Personalized Hey Siri," *Apple Machine Learning Journal*, vol. 1, 9 Apr. 2018. [Online]. Available: https://machinelearning.apple.com/2018/04/16/personalized-hey-siri.html.

[11] M. Wollerton, "Siri vs. Alexa vs. Google Assistant," *CNET*, Aug. 17, 2018. [Online]. Available: https://www.cnet.com/news/siri-vs-alexa-vs-google-assistant/.

[12] Siri Speech Recognition Team, "Finding Local Destinations with Siri's Regionally Specific Language Models for Speech Recognition," *Apple Machine Learning Journal*, vol. 1, 10 Aug. 2018. [Online]. Available: https://machinelearning.apple.com/2018/08/09/regionally-specific-language-models.html.

[13] B. Barrett, "What Amazon Echo and Google Home Do With Your Voice Data," *Wired*, Nov. 24, 2017. [Online]. Available: https://www.wired.com/story/amazon-echo-and-google-home-voice-data-delete/.

[14] A. Ng, "Apple will keep conversations with Siri and HomePod a secret," *CNET*, Jun. 5, 2017. [Online]. Available: https://www.cnet.com/news/apple-will-keep-conversations-with-siri-and-homepod-a-secret/.

[15] Apple Inc., "CloudKit," in *Apple Developer Documentation*. 2018. [Online]. Available: https://developer.apple.com/documentation/cloudkit.

[16] I. Petrov, S. Seru, and S. Petrov, "Home Automation System," *CMBES Proceedings*, vol. 34, no. 1, 2018.

[17] D. J. Wigdor, P. A. Hoover, and K. Hofmeester, "Multi-modal interaction on multi-touch display," pat. US Patent 8,487,888, 2013.

[18] F. Sartor, "Modélisation de l'interopérabilité d'objets communicants et de leur coopération : application à la domotique [Modeling of interoperability of communicating object their cooperation: implementation to home automation]," Thesis, Université de Grenoble,

Jul. 2012. [Online]. Available: https://tel.archives-ouvertes.fr/tel-00748676.

[19] C. Wang, T. Jiang, and Q. Zhang, *ZigBee® network protocols and applications*. Auerbach Publications, 2016.

[20] ZigBee Alliance. (2013). ZigBee Specification FAQ. Archived on June 27, 2013, [Online]. Available: https://web.archive.org/web/20130627172453/http://www.zigbee.org/Specifications/ZigBee/FAQ.aspx.

[21] B. Comrie, *Language universals and linguistic typology: Syntax and morphology*. University of Chicago press, 1989.

[22] ——, "Ergativity," in *Syntactic typology: Studies in the phenomenology of language*. 1978, pp. 329–394.

[23] T. Lucien, "Eléments de syntaxe structurale [Elements of structural syntax]," *Klincksieck, Paris*, 1959.

[24] N. Chomsky, *Topics in the theory of generative grammar*. Walter de Gruyter, 2013, vol. 56.

[25] R. H. Arpaci-Dusseau and A. C. Arpaci-Dusseau, *Operating Systems: Three Easy Pieces*, 0.91. Arpaci-Dusseau Books, 2015.

[26] A. Silberschatz, P. B. Galvin, G. Gagne, and A Silberschatz, *Operating system concepts*. Addison-wesley Reading, 1998, vol. 4.

[27] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120–126, 1978.

[28] S. Goldwasser and S. Micali, "Probabilistic encryption & how to play mental poker keeping secret all partial information," in *Proceedings of the fourteenth annual ACM symposium on Theory of computing*, ACM, 1982, pp. 365–377.

[29] W. Diffie and M. Hellman, "New directions in cryptography," *IEEE transactions on Information Theory*, vol. 22, no. 6, pp. 644–654, 1976.

[30] D. Adrian, K. Bhargavan, Z. Durumeric, P. Gaudry, M. Green, J. A. Halderman, N. Heninger, D. Springall, E. Thomé, L. Valenta, *et al.*, "Imperfect forward secrecy: How Diffie-Hellman fails in practice," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, ACM, 2015, pp. 5–17.

[31] J. Daemen and V. Rijmen, "AES proposal: Rijndael," 1999.

[32] B. Schneier, J. Kelsey, D. Whiting, D. Wagner, C. Hall, N. Ferguson, T. Kohno, and M. Stay, "The Twofish team's final comments on AES Selection," *AES round*, vol. 2, 2000.

[33] G. Irazoqui and X. Guo, "Cache Side Channel Attack: Exploitability and Countermeasures," *Black Hat Asia*, vol. 2017, 2017.

[34] N. Ferguson, J. Kelsey, S. Lucks, B. Schneier, M. Stay, D. Wagner, and D. Whiting, "Improved cryptanalysis of Rijndael," in *International Workshop on Fast Software Encryption*, Springer, 2000, pp. 213–230.

[35] L. Xu, "Securing the enterprise with Intel AES-NI," *Intel Corporation*, 2010.

[36] E. Eldon, "Single sign-on service OpenID getting more usage," *VentureBeat*, Apr. 14, 2009.

[37] P. Crowley. (Jun. 1, 2005). Phishing attacks on OpenID, [Online]. Available: http://lists.danga.com/pipermail/yadis/2005-June/000470.html.

[38] E. Tsyrklevich and V. Tsyrklevich, "Single Sign-On for the Internet: A Security Story," *BlackHat USA*, 2007.