

# From Proprietary to High-Level Trigger-Action Programming Rules: A Natural Language Processing Approach

Ekene Attoh

Web & Information Systems Engineering Lab  
Vrije Universiteit Brussel  
Brussels, Belgium  
eattoh@vub.be

Beat Signer

Web & Information Systems Engineering Lab  
Vrije Universiteit Brussel  
Brussels, Belgium  
bsigner@vub.be

## ABSTRACT

With the rise of popular task automation or IoT platforms such as *If This Then That (IFTTT)*, users can define rules to enable interactions between smart devices in their environment and thereby improve their daily lives. However, the rules authored via these platforms are usually tied to the platforms and sometimes even to the specific devices for which they have been defined. Therefore, when a user wishes to move to a different environment controlled by a different platform and/or devices, they need to recreate their rules for the new environment. The rise in the number of smart devices further adds to the complexity of rule authoring since users will have to navigate an ever-changing landscape of IoT devices. In order to address this problem, we need human-computer interaction that works across the boundaries of specific IoT platforms and devices. A step towards this human-computer interaction across platforms and devices is the introduction of a high-level semantic model for end-user IoT development, enabling users to create rules at a higher level of abstraction. However, many users who already got used to the rule representation in their favourite tool might be unwilling to learn and adapt to a new representation. We present a method for translating proprietary rules to a high-level semantic model by using natural language processing techniques. Our translation enables users to work with their familiar rule representation language and tool, and at the same time apply their rules across different IoT platforms and devices.

## CCS CONCEPTS

• **Human-centered computing** → **Ubiquitous and mobile computing systems and tools.**

## KEYWORDS

Internet of Things, cross-platform IoT, natural language processing, end-user authoring, semantic interoperability

## 1 INTRODUCTION AND RELATED WORK

The interoperability issue between devices of different brands in the domain of the Internet of Things (IoT) is still ever present, a main reason being the unwillingness of major device manufacturers to provide a means for their devices to interoperate with their competitors' devices [16]. Noura et al. [21] studied the interoperability gaps present in state-of-the-art IoT solutions. They defined different categories to classify IoT interoperability issues and found that most IoT solutions do not support *cross-platform* and *cross-domain* interoperability. If supported, these categories can enable

IoT users to exploit different IoT services independently of the platform (e.g. Apple or Samsung) or domain (e.g. health or mobility). End-user development has been suggested as a means to give users control over IoT solutions. According to Barricelli et al. [3], end-user development enables users to develop and adapt systems at a level of complexity that is adequate to their background and skills. Various tools have been proposed to support end-user IoT development [9, 10, 22]. Markopoulos et al. [17] noted that the most common programmatic end-user control of IoT applications is through specifying rules. Previous studies [4, 26] have further shown that a rule-based approach is easily understandable and enables end users to create their own programs. Li et al. [15] identified that although the popular IoT task automation platform IFTTT<sup>1</sup> enables users to create rules across various devices and services, only a certain number of (partner) devices and services are supported.

Corno et al. [5] reiterate this by stating that most end-user development platforms adopt a vendor-centric abstraction, thus requiring that every online service needs to be programmed in a specific way. They argue that this poses interoperability challenges since users need to know any technological details to execute the intended behaviours beforehand. This approach is inadequate in future IoT environments like smart cities, as *things* will not always be known a priori but might dynamically appear and disappear [5].

To address this interoperability issue, Li et al. [15] presented a solution allowing users to author rules for their IoT devices by demonstrating interactions between smart devices using their mobile phones. The solution consists of an *Android* application enabling users to create automation scripts. These scripts are composed by recording the actions users perform on the mobile application of their smart devices. The scripts can then be triggered to perform the actions which were demonstrated by the user, using a source from another application such as a notification from a motion sensor application. We note that since the solution was designed to work on the *Android* operating system, the created automations will only be usable on *Android* devices. This implies that a user would lose all their automations if they were, for example, going to switch to an *iOS* device. Further, only IoT devices whose manufacturers provide an *Android* application for remote control are supported. Corno et al. [6] attempted to address the interoperability issue by introducing the *EUPont* (End User Programming Ontology) high-level semantic model for end-user IoT development. With *EUPont*, users no longer need to create rules for specific devices or services, but they can define abstract rules such that any device or service able to perform the required action can be used to execute those rules. For example, a specific IFTTT rule like *"If my smart sensor X detects*

<sup>1</sup><https://ifttt.com>

that I am home and the outside temperature is less than 10 degrees, then turn on my smart heater H” would have to be recreated if a user switches to a smart sensor “Y” or smart heater “T”. With EUPont, the rule can be transformed to “If I am in an indoor place and the outside temperature is less than 10 degrees, then start heating the indoor place”. Using any platform understanding the EUPont ontology, a user’s rules can be executed on any device that can detect the user’s presence and also heat up their environment. A user study revealed that the EUPont representation allows end users to reduce errors and the time needed to compose their IoT applications.

Another factor contributing to the problem of interoperability in IoT is an issue present in end-user IoT development. As more technologies are supported by platforms such as IFTTT, the design space also grows and it becomes more difficult for users to discover rules and their related functionality [8]. This increased complexity will lead to the creation of multiple rules with similar functionality realised in different possible ways, thus making interoperability more difficult. The use of recommendations in end-user development tools has been proposed to address this issue, similar as for the general development of software artefacts [14, 20].

However, these opportunities for recommendations have not yet been consistently explored to support end-user development but rather focus on supporting professional developers [8]. Therefore, the *TAPrec* end-user development platform supporting the composition of trigger-action rules based on dynamic recommendations has been introduced. At composition time, it suggests new rules to be used or actions, which are based on the rule’s final purpose such as illuminating a place rather than details like device brands and manufacturers, for completing a rule.

Mattioli et al. [18] proposed a solution that suggests relevant triggers, operators and actions to a user during rule composition. The system provides both, step-by-step and full-rule recommendations and a user is either recommended components to complete their rule or the system suggests a complete rule. Jeong et al. [13] introduced a framework to analyse the usage logs of devices in an IoT context and make rule recommendations to users based on the analysis of their device usage patterns. Their solution is further able to make recommendations based on the usage pattern of other users who are in a similar context.

Huang et al. [12] proposed *InstructableCrowd*, a crowd-sourcing system enabling users to create IF-THEN rules based on their needs. Based on a smartphone user interface, users can describe their problems—such as often being late for a meeting—to *crowdworkers*, and the *crowdworkers* can then create rules addressing a user’s needs and send them back to their phone. *HeyTAP* [7] enables users to describe the desired behaviour of their smart devices to their system through conversations (text or voice) and get rule recommendations materialising their stated intentions.

Note that the *situation* concept proposed by Trullemans et al. [25] and implemented in the Context Modelling Toolkit (CMT) is another means to tackle the complexity of authoring rules with similar functionality. They proposed that the *trigger* side of a rule can lead to the definition of a reusable *situation* rather than just triggering an action. For instance, the rule “If my heart rate is higher than 100 bpm” could lead to the situation “My heart rate is high”. A situation can then further be used on the *trigger* side of a new rule definition. This eliminates the need for another user to understand all the

low-level details and they could simply use the situation “My heart rate is high” in their own automations.

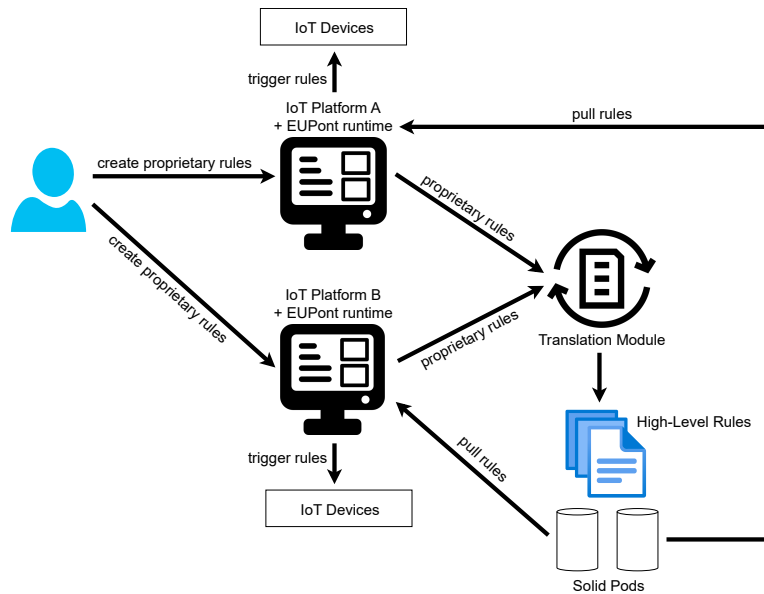
Although the presented related work proposed solutions to address IoT interoperability issues, to the best of our knowledge they only propose solutions focusing on the creation of *new* rules by users. Most research is focused on enabling users to create (new) rules in novel ways and using *new* systems to bridge the interoperability gap rather than enabling users to retain their current tools and methods, while still being able to benefit from solutions that offer cross-platform interoperability [1]. Based on our analysis of related work, we identified two major problems to be addressed:

**Loss of Tooling Choice:** As mentioned before, various solutions have been put forward to bridge the cross-platform gap [6, 15], but they also propose the use of new tools and languages. This means that users will need to learn to use new tools and rule descriptions in order to create rules that can be used across different platforms.

**Rule Authoring Complexity:** Related work further shows that due to the rise in the number of smart devices, the discoverability of rules and their related functionality becomes more complex [8]. Therefore, users do not only need to use new tools and rule description languages to benefit from cross-platform interoperability solutions, but they also have to navigate an ever-changing landscape of IoT devices and services while authoring their rules. This additional complexity may not only create an entry barrier for new users, but also increase a user’s time needed to create their desired automation. We analysed the IFTTT user recipes (rules) from the May 2017 dataset of Mi et al. [19] and found that out of the total 279’828 user recipes, there were 863 duplicate triggers (total number of triggers that were used more than once) and 502 duplicate actions (total number of actions that were used more than once). The identification of these triggers and actions, and the understanding of the underlying functionality is not a very complex task if they are relatively simple, but can easily become more difficult for more complicated or lesser-known triggers and actions. Dealing with that complexity in order to enjoy cross-platform intelligibility might be unmanageable for most users and in particular for non-expert users.

## 2 SOLUTION

We propose a Natural Language Processing (NLP) approach for automatically translating proprietary end-user rules to the EUPont high-level abstraction by Corno et al. [6]. This method provides end users with a *Write Once, Run Anywhere* paradigm where a user can retain the authoring tool and language description of their choice but has the flexibility to use their rules across different platforms. A user simply has to write their rule as they would normally and have it translated to an equivalent EUPont representation. For instance, let us assume that a user has previously composed the IFTTT rule “If AC brand X is turned off, then activate my camera brand Y” for their smart home. They now find themselves on vacation in a smart environment which uses an air conditioner (AC) of brand Z and a camera of brand C. With current solutions, this user needs to create a new rule “If AC brand Z is turned off, then activate my camera brand C” in order to have the same experience in their vacation



**Figure 1: Proprietary IoT rule translation technique**

environment as they would enjoy at their home. Instead, we propose a solution where a rule can be automatically translated to the EUPont generalisation “*If device turned off, then connect to device*”. Just as the JSON<sup>2</sup> format is generic such that most (modern) language compilers and interpreters are equipped with JSON parsers in order to work with JSON data, the intention behind the EUPont representation is similar such that IoT platforms with the EUPont “*runtime*” might be able to work with the representation. Therefore, a proprietary rule such as an IFTTT rule has to be written only *once* and can be translated to the EUPont representation in order to be used across different platforms. This means that an EUPont-powered platform can make it possible for any device which might be triggered off to be used as the *trigger* of the rule. For the rule’s action, a camera can be mapped to the high-level action “*Connect to device*”, which can then be triggered when the rule is executed. A user is therefore not limited to using devices of brand X or Y in order to take advantage of their already composed rule.

An overview of our proposed IoT rule translation approach is provided in Figure 1. A user is able to create (proprietary) IoT rules, which can be seen as the *Write Once* part, using any platform of their choice (e.g. IoT Platform A or IoT Platform B). These platforms have access to our translation approach described in the remaining part of this paper and the Translation Module then converts the created rules to the high-level EUPont representation. As stated previously, a user would need to duplicate and further customise a rule authored to work on a specific platform in order to use that rule on a different platform, given that each platform stores its users’ data locally. In order to address this issue, we also introduce the use of Solid Pods to store the automatically translated high-level rules. Solid aims to provide data independence as well as simple yet powerful data management mechanisms. Using Solid, applications no longer store their data themselves but request

access to retrieve data from users’ Pods [24]. The Solid integration thus enables a user to use these high-level rules on any IoT platform by granting individual platforms access to their Pod. This, in combination with the translation step, leads to the *Run anywhere* part of our proposed solution. A user can write their rules using a proprietary platform such as IFTTT, these rules are then translated by our Translation Module and stored centrally on the user’s Solid Pod. A platform can then pull the user’s translated rules from the Solid Pods and with the help of the EUPont runtime, trigger the required actions on the corresponding IoT devices. This also puts users in control of their data, with their Solid Pods as the single access point and source of truth concerning their IoT rules.

Ultimately with our proposed translation solution, we aim to minimise or even eliminate a user’s need to search for and/or understand the equivalent EUPont representation for their proprietary rules. In order to achieve this goal, we first propose that the user should be kept in the loop such that initially, they might manually select the best matching translation in situations where the one proposed by the system is inadequate. With this method, we intend that the best results are learned over time and then proposed to the user. Due to the popularity of the IFTTT platform, we have selected IFTTT rules as the first type of input to be addressed by our translation method. In future work, we intend to apply our approach to other IoT platforms such as Home Assistant<sup>3</sup>.

As mentioned earlier, the May 2017 dataset of Mi et al. [19] contains 279’828 user recipes, implying that each of those recipes would need to be created for each new platform it is used on. With our proposed solution, these recipes can not only be written just once and then be automatically translated to run anywhere, but the author of a recipe can use their preferred authoring tool and keep the ownership of their data (rules). In the remainder of this paper, we describe the translation approach depicted as Translation Module in Figure 1.

<sup>2</sup><https://www.json.org>

<sup>3</sup><https://www.home-assistant.io>

## 2.1 Dataset

Mi et al. [19] collected published IFTTT recipes (rules) from November 2016 until May 2017. For our automatic translation of rules, we decided to use the most recent May 2017 dataset containing a total of 279'828 recipes (rules). Note that not each of the collected recipes does necessarily contain unique triggers and actions. For example, the trigger “Any new photo by you” is used 9680 times in this dataset. Therefore, the 279'828 rules of the collected dataset consist of a total of 1017 different triggers and 616 different actions of which 154 triggers and 114 actions appeared only *once*. For our proof-of-concept implementation and evaluation, we ran our solution on all different triggers and actions.

## 2.2 Data Preparation

Before applying our translation technique, we performed some cleaning of the data in order to remove any present anomalies and to prepare it to be used for the translation steps. For the recipes (rules) present in Mi et al.'s [19] dataset, we noticed that some of the triggers and actions contained a forward slash character. We thus removed that character from the triggers and actions and further separated the triggers and actions into two different lists. In the following, we refer to these lists as the *IFTTT dataset*. We also transformed the ontology proposed by Corno et al. [6] from XML to JSON in order to extract its high-level triggers and actions. We removed the *Trigger* and *Action* suffixes from the triggers and actions as they were redundant and separated the high-level trigger and action names into two different lists. In the following, we refer to these lists as the *EUPont dataset*.

## 2.3 Translation Technique

Our aim in translation was to take a rule written by a user in a proprietary format (e.g. IFTTT) and return a generalisation of that rule in the high-level EUPont format by Corno et al. [6] that is as accurate as possible. This would enable the user to maintain their use of the IFTTT platform without having to learn a new rule description language. In order to perform this automatic translation from proprietary rules to high-level rules, we apply some natural language processing (NLP) techniques. According to Quarashi et al. [23], measuring text similarity is an important part of NLP applications, such as information retrieval, machine translation and text summarisation. In order to perform the translation, we applied different *document similarity* algorithms to both, the IFTTT and EUPont datasets, using the algorithm shown in Listing 1.

```

1 for each trigger x in EUPont dataset:
2   for each trigger y in IFTTT dataset:
3     run document_similarity(x,y)
4 return x, y, Similarity(x,y)
5 order by similarity descending
6
7 for each action a in EUPont dataset:
8   for each action b in IFTTT dataset:
9     run document_similarity(a,b)
10 return a, b, Similarity(a,b)
11 order by similarity descending

```

**Listing 1: Translation algorithm pseudocode**

We used the algorithm with three implementations (spaCy, AllenNLP and combined similarity) of the `document_similarity(x,y)` function in Listing 1 and compared the results as described.

**spaCy Similarity:** *spaCy*<sup>4</sup> is a free open source Python library for advanced Natural Language Processing. It can be used to build information extraction, natural language understanding systems or even to pre-process text for deep learning. We use *spaCy*'s *similarity* feature to compare how similar a given IFTTT trigger and action are to the high-level triggers and actions presented in the ontology by Corno et al. [6]. We then return the IFTTT trigger or action name together with the computed similar EUPont trigger or action names, as well as the corresponding similarity level.

```

1 [
2   {
3     "Every Time": {
4       "ifttt_name": "Any event starts",
5       "similarity": 0.7474772725000891 }
6   },
7   {
8     "Every Day": {
9       "ifttt_name": "Any event starts",
10      "similarity": 0.7034427432691928 }
11  },
12  {
13    "Every Week": {
14      "ifttt_name": "Any event starts",
15      "similarity": 0.6832991463006063 }
16  },
17  {
18    "Every Year": {
19      "ifttt_name": "Any event starts",
20      "similarity": 0.6819517479034135 }
21  }
22 ]

```

**Listing 2: spaCy IFTTT trigger translation example**

An example of a result is shown in Listing 2, where the first entry has the EUPont trigger “Every Time” returned by the *spaCy* approach for the IFTTT trigger name “Any event starts”. Since not all translations returned by the algorithm are relevant, we defined a *threshold value* which is used to filter out results whose similarity falls below that value. While the threshold is customisable, based on our initial analysis we set its value to 0.55.

**AllenNLP Similarity:** AllenNLP [11] is an entire platform for solving NLP tasks and comes with a Python library. We applied the *textual entailment* feature of AllenNLP which, for a pair of sentences, predicts whether the facts in the first sentence imply the facts in the second. We thus determine the textual entailment between each IFTTT trigger and action in the dataset, and the high-level EUPont [6] triggers and actions. The AllenNLP textual entailment algorithm returns *entailment* (a measure of the similarity of both texts), *contradiction* (a measure of the dissimilarity of both texts) and *neutral* (a measure of the neutrality of both texts). We return the IFTTT triggers and actions together with the computed similar EUPont triggers and actions as well as their entailment, contradiction and neutral values as illustrated in Listing 3.

**Combined Similarity:** Our preliminary analysis revealed that the *spaCy* approach returns more reliable results than the AllenNLP approach. In order to improve the translation results and reduce any noise, we defined a new approach where the AllenNLP algorithm is used to compare the similarity between the initial *spaCy* results and the EUPont triggers and actions. For example, for the IFTTT trigger “Any event starts”, we see that the first *spaCy* result returned is “Every Time” while the first AllenNLP result returned is “Taken”.

<sup>4</sup><https://spacy.io/usage/spacy-101>

```

1 [
2   [
3     {
4       "ifttt_name": "Any event starts",
5       "eupont_hypothesis": "Taken",
6       "allen_nlp_entailment": 92.16328263282776,
7       "allen_nlp_contradiction": 3.0818356201052666,
8       "allen_nlp_neutral": 4.75488156080246
9     },
10    {
11      "ifttt_name": "Any event starts",
12      "eupont_hypothesis": "Received",
13      "allen_nlp_entailment": 91.5201187133789,
14      "allen_nlp_contradiction": 3.172384202480316,
15      "allen_nlp_neutral": 5.307500064373016
16    },
17    {
18      "ifttt_name": "Any event starts",
19      "eupont_hypothesis": "Temporal",
20      "allen_nlp_entailment": 88.79325985908508,
21      "allen_nlp_contradiction": 5.660351365804672,
22      "allen_nlp_neutral": 5.546396225690842
23    },
24    {
25      "ifttt_name": "Any event starts",
26      "eupont_hypothesis": "Received From Diy",
27      "allen_nlp_entailment": 86.96905374526978,
28      "allen_nlp_contradiction": 2.1996214985847473,
29      "allen_nlp_neutral": 10.831315815448761
30    }
31  ]
32 ]

```

Listing 3: AllenNLP IFTTT trigger translation example

While the first result returned by the spaCy approach might be acceptable, the result returned by the AllenNLP approach is not. However, while conducting our preliminary analysis, we noted that there were more accurate matches further down in the list of results returned by the spaCy approach. We therefore ran the AllenNLP algorithm using the preliminary spaCy results and the EUPont triggers in order to further improve the translation.

In Listing 4 we can see that the first result returned using this combined approach is *“Started Activity”*. This result is obtained by combining (averaging) its original spaCy similarity value (61.39) with the similarity value obtained when using the AllenNLP algorithm (85.80). The entry therefore has a combined similarity of 73.60. We are thus able to move this specific spaCy result—which had initially a low ranking compared to the initial *“Every Time”* top result—to the top. We perform this process for each result returned by the spaCy approach and therefore the results with a high combined similarity are most likely to be the most accurate since they have both, high spaCy and high AllenNLP similarities. An example of the result obtained using this combined approach is highlighted in Listing 4.

While there are several NLP libraries available for the Python programming language such as scikit-learn<sup>5</sup> and PyTorch<sup>6</sup>, we opted for the spaCy<sup>7</sup> and AllenNLP libraries mainly due to their user friendliness. spaCy is also a popular choice in NLP tasks given its speed and the ease with which it lets users build solutions. Similarly, AllenNLP enjoys popularity due to the speed and ease with which it lets a user build prototypes.

<sup>5</sup>[https://scikit-learn.org/stable/tutorial/text\\_analytics/working\\_with\\_text\\_data.html](https://scikit-learn.org/stable/tutorial/text_analytics/working_with_text_data.html)

<sup>6</sup><https://pytorch.org/readthedocs.io/en/latest/>

<sup>7</sup><https://spacy.io>

```

1 [
2   {
3     "ifttt_name": "Any event starts",
4     "eupont_hypothesis": "Started Activity",
5     "spacy_similarity": 61.39593233371522,
6     "allen_nlp_entailment": 85.80678701400757,
7     "allen_nlp_contradiction": 3.3948026597499847,
8     "allen_nlp_neutral": 10.798408836126328,
9     "combined_similarity": 73.6013596738614
10  },
11  {
12    "ifttt_name": "Any event starts",
13    "eupont_hypothesis": "Position Registration",
14    "spacy_similarity": 57.52355435396419,
15    "allen_nlp_entailment": 81.54605627059937,
16    "allen_nlp_contradiction": 5.951366946101189,
17    "allen_nlp_neutral": 12.50256896018982,
18    "combined_similarity": 69.53480531228178
19  },
20  {
21    "ifttt_name": "Any event starts",
22    "eupont_hypothesis": "Device Turned On",
23    "spacy_similarity": 56.46723924755545,
24    "allen_nlp_entailment": 80.1530659198761,
25    "allen_nlp_contradiction": 5.053842067718506,
26    "allen_nlp_neutral": 14.793087542057037,
27    "combined_similarity": 68.31015258371578
28  },
29  {
30    "ifttt_name": "Any event starts",
31    "eupont_hypothesis": "Time",
32    "spacy_similarity": 67.30982538675227,
33    "allen_nlp_entailment": 65.08164405822754,
34    "allen_nlp_contradiction": 16.626055538654327,
35    "allen_nlp_neutral": 18.29230487346649,
36    "combined_similarity": 66.1957347224899
37  }
38 ]

```

Listing 4: Combined IFTTT trigger translation example

### 3 RESULTS

We recorded and compared the results we obtained when applying each of the three different approaches described in Section 2.3 on recipes of the Mi et al. [19] dataset. With our translation technique, we intended that the first result recommended by each approach should produce the most accurate high-level EUPont generalisation of the IFTTT triggers and actions. However, in situations where this is not the case, a user should at least be able to find an accurate high-level EUPont generalisation within the first five returned results. We decided to consider only the first five results by each approach to reduce the potential burden a user might face when looking for the best result. In our results, we mark an entry with *“No result”* if no suitable match has been returned as part of the first five results. For trigger or action names that we consider to be ambiguous—any trigger or action whose meaning could have multiple interpretations—we mark the entry and the resulting translations as *“Ambiguous”*. For instance, the trigger *“Air quality changed”* is marked as *“Ambiguous”* because the change in air quality could either be positive or negative. Therefore one approach might return *“Air quality decreased”* as its best result, while another approach might return *“Air quality increased”* as its best result. We further note that several possible acceptable results were returned for certain IFTTT triggers and actions by our approach in the different columns.

No.	IFTTT Name	spaCy 1	spaCY 2	AllenNLP 1	AllenNLP 2	Combined 1	Combined 2
1	...	...	...	...	...	...	...
2	A C <sup>8</sup> turned off	Device Turned Off	Device Turned Off (1)	Brightness Decreased	No result	Device Turned Off	Device Turned Off (1)
23	Action Button Pressed	Tap Button Activity	Tap Button Activity (1)	Taken	No result	Tap Button Activity	Tap Button Activity (1)
34	Air filter needs cleaning	Air Purifier Enabled	No result	Moving	No result	Started Cleaning	Sensed Air Quality Decreased (3)
35	Air pressure drops below	Increased Air Pressure	Sensed Air Pressure Decreased (4)	Temporal	No result	Sensed Air Pressure Decreased	Sensed Air Pressure Decreased (1)
36	Air pressure rises above	Increased Air Pressure	Increased Air Pressure (1)	Temporal	No result	Sensed Air Pressure Increased	Sensed Air Pressure Increased (1)
37	Air purifier is turned on	Air Purifier Enabled	Device Turned On (4)	Taken Audio	Device Turned On (2)	Device Turned On	Device Turned On (1)
38	Air quality changed (ambiguous)	Increased Air Quality	Ambiguous	Temporal	Ambiguous	Increased Air Quality	Ambiguous
48	An animal has been seen outside	Received Like	No result	Taken	Taken Image (2)	Taken	Taken Image (2)

Table 1: IFTTT trigger translation results

No.	IFTTT Name	spaCy 1	spaCY 2	AllenNLP 1	AllenNLP 2	Combined 1	Combined 2
1	...	...	...	...	...	...	...
10	Add a file	Share File	Save File (2)	Information	No result	Share File	Save File (2)
11	Add a new site (ambiguous)	Connect To Web Service	Ambiguous	Start Focusing	Ambiguous	Save Media Information	Ambiguous
17	Add a private bookmark	Save Web Bookmark	Save Web Bookmark (1)	Information	No result	Get	Save Media Information (4)
31	Add item to Reading List	Add Calendar Item	Add Remind (2)	Information	Save (2)	Share Post	Add Remind (3)
32	Add item to your feed	Add Calendar Item	No result	Start Cooking	No result	Get	Save Media Information (2)
33	Add message	Send Message	No result	Information	No result	Add Reminder	No result
34	Add mix to favorites	Add Remind	No result	Get	No result	Add Remind	No result
35	Add new contact to list	Send To Display	Save Contact (5)	Information	No result	Save Media Information	Save Media Information (1)

Table 2: IFTTT action translation results

We split our findings into two different tables, with Table 1 showing the results for triggers and Table 2 highlighting the results for actions. Each table contains the following columns:

- **No.:** Entry Number
- **IFTTT Name:** IFTTT trigger or action name
- **spaCy 1:** EUPont trigger or action name with the highest similarity value using the spaCy similarity algorithm.
- **spaCY 2:** EUPont trigger or action name most accurately representing the IFTTT trigger or action name using the spaCy similarity algorithm (position in the result list in brackets).
- **AllenNLP 1:** EUPont trigger or action name with the highest similarity value using the AllenNLP text entailment algorithm.
- **AllenNLP 2:** EUPont trigger or action name most accurately representing the IFTTT trigger or action name using the AllenNLP text entailment algorithm (position in the result list in brackets).
- **Combined 1:** EUPont trigger or action name with the highest similarity value using the combined approach.
- **Combined 2:** EUPont trigger or action name most accurately representing the IFTTT trigger or action name using the combined approach (position in the result list in brackets).

For the results presented in this paper, we randomly selected 50 triggers and actions from the results we obtained from running our approaches on the dataset. The results from these 50 triggers and actions were then manually analysed (e.g. to identify and label the most accurate EUPont triggers and actions) in order to populate the entries in Table 1 and Table 2. Note that only a few representative entries from these two tables are shown but the entire tables as well as the complete but non-annotated results of the presented approaches are available online [2].

### 3.1 Analysis

We sought to determine which out of the three approaches is performing best. We consider an approach to be performing better than another approach based on a combination of the following criteria:

- It has more top results than the other approach
- It has more top 5 results than the other approach
- It has fewer cases where a translation could not be found in the top 5 results compared to the other approach

Approach	First Result	Top Five Result	No result
spaCy	13	9	16
AllenNLP	2	12	24
Combined	16	13	9

Table 3: Summary of trigger translation results

Approach	First Result	Top Five Result	No result
spaCy	10	8	22
AllenNLP	1	11	28
Combined	8	19	13

Table 4: Summary of action translation results

In our analysis, entries which were marked as *ambiguous* (12 out of the 50 randomly selected triggers) were not considered. Therefore, we found that for the remaining 38 triggers, our combined approach returned the best EUPont match as the first result for 16 of those triggers as summarised in Table 3. For 13 of the triggers, the best EUPont match was not the first result but could be found in the top 5 results. However, for 9 of the IFTTT triggers, a suitable EUPont match could not be found by our combined approach. Using the spaCy approach, we found that the best EUPont match was returned as the first result for 13 triggers, while for 9 triggers, the best EUPont match was not the first result but could be found in the top 5 results. However, for 16 triggers, a suitable EUPont match could not be found. Similarly, using the AllenNLP approach we see that the best EUPont match was returned as the first result for 2 triggers. For 12 of the triggers, the best EUPont match was not the first result but could be found in the top 5 results, while for 24 of the triggers no suitable EUPont match could be found.

For the actions, there were 10 entries marked as *ambiguous*. Table 4 shows that when using our combined approach on the 40 considered actions, for 8 of the actions the best EUPont match was returned as the first result, while for 19 of the actions the best EUPont match was not the first result but could be found in the top 5 results and for 13 of the actions, no suitable EUPont match could be found. Using the spaCy approach, 10 of the actions had the best EUPont match returned as the first result, while for 8 of the actions the best EUPont match was not the first result but could be found in the top 5 results. However, for 22 of the actions, a suitable EUPont match could not be found. Using the AllenNLP approach, we see that only for 1 out of the actions the best EUPont match was returned as the first result, while for 11 of the actions the best EUPont match was not the first result but could be found in the top 5 results and for 28 of the actions, no suitable EUPont match could be found.

Based on these results, we can conclude that our combined approach is the best-performing approach using our test dataset for both triggers and actions. For triggers, our combined approach returns the highest number of top results, top 5 results and has the smallest number of cases where no result has been returned as part of the top 5. For the actions though, the spaCy approach returns more top results than our combined approach. However, spaCy returns significantly fewer top 5 results and has a larger number of cases where no result was returned as part of the top 5.

## 4 EVALUATION

### 4.1 EuPont Evaluation

Corno et al [6] conducted a user study to evaluate the suitability and the understandability of the EUPont approach by end-users. The study was a controlled in-lab experiment that involved 30 participants, 15 of whom only had programming experience. It focused on the creation of IoT applications both with the current low-level representation of IFTTT and the high-level representation of EUPont. The study addressed the research questions “Does the EUPont representation help users create their IoT applications more effectively and efficiently compared to the low-level representation?” and “Which of the two representations is preferred by users, and which are the perceived advantages and disadvantages of the two solutions?”. In summary, the results of the study successfully demonstrated that the EUPont representation allowed end-users to reduce the errors and time needed to compose their IoT applications, and introduced numerous benefits in terms of understandability and ease of use.

### 4.2 User Evaluation

Based on the findings of the user study conducted by Corno et al [6], we consider the EUPont representation to be a suitable high-level representation of IoT rules for end users. To further evaluate our approach and gather some feedback for future work, we conducted a preliminary survey targeting a number of respondents who were already familiar with the use of IoT automation solutions. With this survey, we aimed to investigate whether real IoT users would find the results returned by any of our methods to be good high-level generalisations of the IFTTT triggers and actions included in the dataset described earlier in Section 2. The research question we sought to answer with this preliminary survey was “Are the EUPont translations returned by our methods acceptable to end users?” There were 11 survey respondents who started the survey but only two of them completed the survey. We thus base our analysis on the responses of these two respondents who were both male, aged between 20 and 39 years and have obtained a Master’s degree. In our survey, the respondents were presented a series of IFTTT triggers and actions with their corresponding translations based on the three approaches described in Section 2. They then had to select which method they thought returned a good high-level generalisation of the IFTTT trigger or action and also specify to which degree they found that generalisation to be accurate. They could further select the “N/A” option in case they found that none of the

methods returned a suitable generalisation. For example, given the IFTTT trigger “*New photo upload on page*”, users were presented the following four options:

- Method 1: “Shared Profile Update”
- Method 2: “No Result”
- Method 3: “Shared Post”
- “N/A”

In a follow-up question, they were asked “*To which degree is your chosen method an accurate generalisation of the IFTTT trigger?*” and had the choice of “*Not at all accurate*”, “*Low accuracy*”, “*Accurate*” and “*Very accurate*”.

For situations where multiple methods returned the same value, the respondents were asked to pick any of those methods if they considered the value to be an accurate generalisation of the trigger or action. For example, for the IFTTT trigger “*If new post from search ...*”, method 1 and method 3 returned “*If shared post ...*” and therefore the respondents could choose any of those two methods if they found it a good high-level generalisation of the IFTTT trigger. We followed the same principle as described in Section 2.3 in considering only the first five results of each method.

The triggers and actions selected for the survey were those we consider to be popular in the dataset; that is they were used 1000 or more times in the dataset described in Section 2. Triggers and actions for which the three methods returned no suitable or ambiguous results were not selected. The survey thus comprised questions for 31 triggers and 33 actions. There were 20 out of the 31 triggers and 19 out of the 33 actions where more than one method returned the same value. We will refer to these cases as *triggers with the same result* and *actions with the same result* respectively. For 11 triggers and 14 actions none of the three methods returned the same result. We will refer to these cases as *triggers with different results* and *actions with different results* respectively.

For the 20 triggers with the same result, in 16 cases both respondents selected that result as a good high-level description, while for the other 4 cases they indicated that there was no suitable generalisation. In the case of the 19 actions with the same result, both respondents selected that result as a good high-level description in 14 cases, while for the other 5 cases they indicated that there was no suitable generalisation. For the 11 triggers with different results, there were five triggers where our combined method’s result was not selected. For those five triggers, the result returned by the spaCy method was selected by both respondents; however, none of the other two methods returned a result for those five triggers. There were 5 of the 11 triggers with different results where the result of the combined method was selected by the respondents. Both respondents selected the result returned by the combined method in 3 out of those 5 cases, while for the other 2 cases, only one respondent selected the result returned by the combined method. The second respondent indicated that no suitable generalisation was found. For 3 out of these 4 triggers, the other methods did not return any suitable result. For one of the 11 triggers with different results, the result from our combined method was selected by one respondent while the result from the spaCy method was selected by the second respondent. Finally, there was one trigger where no method’s result was selected even though our combined method was the only method that returned a result for that particular trigger.

In the case of the 14 actions with different results, there were 7 actions where our combined method’s result was not selected. For those 7 actions, the result returned by the spaCy method was selected in 6 cases by both respondents and the result returned by the AllenNLP method was selected once by only one respondent. Our combined method only returned a result for 3 of those seven actions while AllenNLP returned a result for only one of the 7 actions. There were 4 of the 14 actions with different results where the result returned by the combined method was selected by the respondents. Both respondents selected the result returned by the combined method in 2 out of those 4 cases, while for the other 2 cases only one respondent selected the result returned by the combined method. The second respondent indicated that no suitable generalisation was found. For 3 out of these 4 actions, the other methods did not return any suitable result. For 2 of the 14 actions with different results, the result from our combined method was selected by one respondent while the result by the spaCy method was selected by the second respondent. Finally, there was one action where no method’s result was selected even though our combined method was the only method that returned a result for that particular action. In summary, we can conclude that for 24 out of the 31 triggers and 22 out of the 33 actions, both respondents selected the result returned by one of our methods as a good high-level generalisation. For the remaining cases, both respondents did not select the result returned by one of our methods as a good high-level generalisation.

## 5 CONCLUSIONS AND FUTURE WORK

We presented a *Write Once Run Anywhere* paradigm for end-user authoring in IoT settings, helping users to maintain their preferred authoring tool as well as their preferred description language when defining IoT rules that will work across different IoT platforms. In order to achieve this, we employed the use of natural language processing techniques to automatically translate proprietary rules to high-level EUPont rules. The findings of Corno et al [6] demonstrate that end users find the EUPont representation to improve the rule authoring process. We thus used two popular NLP algorithms in two different approaches and proposed a third novel approach by combining these two algorithms and carefully analysed the results obtained from all three approaches. From the results of our analysis in Section 3.1, we see that all three methods return good high-level generalisations with the combined method performing better as described in 3.1 than the other two methods for the given dataset. We acknowledge that only a small set of users completed our survey, but from these preliminary results we see that real IoT users also selected the results returned by our three approaches as good high-level generalisation for the triggers and actions that they were presented. We consider these results to be positive and for future work, we will investigate how to best consistently return the most accurate high-level generalisation for a user’s rules, by either using one or a combination of the methods we have presented. We will also focus on improving the accuracy of the results such that more often the first result returned is the most accurate high-level generalisation and the number of cases where no result is returned is significantly minimised or even completely eliminated. Further, we plan to investigate how our solution can best be integrated



into existing IoT platforms as shown earlier in Figure 1 in order to further evaluate the proposed NLP-based rule translation approach by end users.

## REFERENCES

- [1] Ekene Attoh and Beat Signer. 2021. A Middleware for Implicit Human-Computer Interaction Across IoT Platforms. In *Adjunct Proceedings of UbiComp 2021*. <https://doi.org/10.1145/3460418.3479311>
- [2] Ekene Attoh and Beat Signer. 2023. *Transforming Proprietary to High-Level Trigger-Action Programming Rules Dataset*. <https://doi.org/10.5281/zenodo.10033916>
- [3] Barbara Rita Barricelli et al. 2019. End-User Development, End-User Programming and End-User Software Engineering: A Systematic Mapping Study. *Journal of Systems and Software* 149 (2019). <https://doi.org/10.1016/j.jss.2018.11.041>
- [4] Federico Cabitza et al. 2017. Rule-based Tools for the Configuration of Ambient Intelligence Systems: A Comparative User Study. *Multimedia Tools and Applications* 76, 4 (2017). <https://doi.org/10.1007/s11042-016-3511-2>
- [5] Fulvio Corno, Luigi De Russis, and Alberto Monge Roffarello. 2021. Devices, Information, and People: Abstracting the Internet of Things for End-User Personalization. In *Proceedings of IS-EUD 2021*. [https://doi.org/10.1007/978-3-030-79840-6\\_5](https://doi.org/10.1007/978-3-030-79840-6_5)
- [6] Fulvio Corno, Luigi De Russis, and Alberto Monge Roffarello. 2019. A High-Level Semantic Approach to End-User Development in the Internet of Things. *International Journal of Human-Computer Studies* 125 (2019). <https://doi.org/10.1016/j.ijhcs.2018.12.008>
- [7] Fulvio Corno, Luigi De Russis, and Alberto Monge Roffarello. 2020. HeyTAP: Bridging the Gaps Between Users' Needs and Technology in IF-THEN Rules via Conversation. In *Proceedings of AVI 2020*. <https://doi.org/10.1145/3399715.3399905>
- [8] Fulvio Corno, Luigi De Russis, and Alberto Monge Roffarello. 2020. TAPrec: Supporting the Composition of Trigger-Action Rules Through Dynamic Recommendations. In *Proceedings of IUI 2020*. <https://doi.org/10.1145/3377325.3377499>
- [9] Joëlle Coutaz and James L. Crowley. 2016. A First-Person Experience with End-User Development for Smart Homes. *IEEE Pervasive Computing* 15, 2 (2016). <https://doi.org/10.1109/MPRV.2016.24>
- [10] Giuseppe Desolda, Carmelo Ardito, and Maristella Matera. 2016. End-User Development for the Internet of Things: EFESTO and the 5W Composition Paradigm. In *Proceedings of RMC 2016*. [https://doi.org/10.1007/978-3-319-53174-8\\_5](https://doi.org/10.1007/978-3-319-53174-8_5)
- [11] Matt Gardner et al. 2017. AllenNLP: A Deep Semantic Natural Language Processing Platform. In *Proceedings of NLP-OSS*. <https://doi.org/10.18653/v1/W18-2501>
- [12] Ting-Hao Kenneth Huang, Amos Azaria, and Jeffrey P Bigham. 2016. Instructable-Crowd: Creating IF-THEN Rules via Conversations With the Crowd. In *Proceedings of CHI 2016 EA*. <https://doi.org/10.1145/2851581.2892502>
- [13] Hanjo Jeong et al. 2019. Big Data and Rule-based Recommendation System in Internet of Things. *Cluster Computing* 22, 1 (2019). <https://doi.org/10.1007/s10586-017-1078-y>
- [14] Jacob Krüger. 2018. When to Extract Features: Towards a Recommender System. In *Proc of ICSE 2018 Companion*.
- [15] Toby Jia-Jun Li et al. 2017. Programming IoT Devices by Demonstration Using Mobile Apps. In *Proceedings of IS-EUD 2017*. [https://doi.org/10.1007/978-3-319-58735-6\\_1](https://doi.org/10.1007/978-3-319-58735-6_1)
- [16] Carmelo Fabio Longo et al. 2022. Towards Ontological Interoperability of Cognitive IoT Agents Based on Natural Language Processing. *Intelligenza Artificiale* 16, 1 (2022). <https://doi.org/10.3233/IA-210125>
- [17] Panos Markopoulos, Jeffrey Nichols, Fabio Paternò, and Volkmar Pipek. 2017. End-User Development for the Internet of Things. *TOCHI* 24, 2 (2017). <https://doi.org/10.1145/3054765>
- [18] Andrea Mattioli and Fabio Paternò. 2021. Recommendations for Creating Trigger-Action Rules in a Block-based Environment. *Behaviour & Information Technology* 40 (2021). <https://doi.org/10.1080/0144929X.2021.1900396>
- [19] Xianghang Mi et al. 2017. An Empirical Characterization of IFTTT: Ecosystem, Usage, and Performance. In *Proceedings of IMC 2017*. <https://doi.org/10.1145/3131365.3131369>
- [20] Anh Tuan Nguyen et al. 2016. API Code Recommendation Using Statistical Learning From Fine-Grained Changes. In *Proceedings of FSE 2016*. <https://doi.org/10.1145/2950290.2950333>
- [21] Mahda Noura, Mohammed Atiquzzaman, and Martin Gaedke. 2019. Interoperability in Internet of Things: Taxonomies and Open Challenges. *Mobile Networks and Applications* 24, 3 (2019). <https://doi.org/10.1007/s11036-018-1089-9>
- [22] Bauyrzhan Ospan et al. 2018. Context Aware Virtual Assistant With Case-based Conflict Resolution in Multi-User Smart Home Environment. In *Proceedings of CoCoNet 2018*. <https://doi.org/10.1109/CoCoNet.2018.8476898>
- [23] Abdul Wahab Qurashi, Violeta Holmes, and Anju P Johnson. 2020. Document Processing: Methods for Semantic Text Similarity Analysis. In *Proceedings of INISTA 2020*. <https://doi.org/10.1109/INISTA49547.2020.9194665>
- [24] Andrei Vlad Sambra et al. 2016. *Solid: A Platform for Decentralized Social Applications Based on Linked Data*. Technical Report. MIT CSAIL & Qatar Computing Research Institute.
- [25] Sandra Trullemans, Lars Van Holsbeeke, and Beat Signer. 2017. The Context Modelling Toolkit: A Unified Multi-layered Context Modelling Approach. *PACMHCI* 1, EICS (2017). <https://doi.org/10.1145/3095810>
- [26] Blase Ur et al. 2016. Trigger-Action Programming in the Wild: An Analysis of 200,000 IFTTT Recipes. In *Proceedings of CHI 2016*. <https://doi.org/10.1145/2858036.2858556>