D. Ferraris, C. Fernandez-Gago, and J. Lopez, "POM: A Trust-based AHP-like Methodology to Solve Conflict Requirements for the IoT", *Collaborative Approaches for Cyber Security in Cyber-Physical Systems*, pp. 145-170, 2023.
http://doi.org/https://doi.org/10.1007/978-3-031-16088-2_7
NICS Lab. Publications: https://www.nics.uma.es/publications

POM: A Trust-based AHP-like Methodology to Solve Conflict Requirements for the IoT

Davide Ferraris, Carmen Fernandez-Gago, and Javier Lopez

Network, Information and Computer Security Lab, University of Malaga, 29071 Malaga, Spain
{ferraris,mcgago,jlm}@lcc.uma.es

Abstract. The Internet of Things (IoT) is an environment of interconnected entities, which are identifiable, usable and controllable via the Internet. Trust is necessary in a system such as the IoT as the entities involved should know the other entities they have to interact with. Trust management systems can help entities connected to others in the decision-making process. In order to guarantee trust in an IoT entity, it is useful to consider it during all its System Development Life Cycle (SDLC) especially in the earliest phases. The requirements phase is one of the first and most important phases and trust requirements must be elicited in order to guarantee that the built entity can be trusted. During this phase, it is possible to develop conflicting requirements due to conflicting needs or functionalities. Decision-making processes can be helpful in order to solve these issues. The Analytic Hierarchy Process (AHP) is a discipline that supports decision-makers in choosing between heterogeneous and conflicting alternatives, but it has several issues especially if there are numerous parameters. Thus, we propose an AHP-like methodology called Pairwise Ordination Method (POM) useful to solve issues arisen among conflicting requirements deciding which one is the less important in order to change or delete it, maximising the trust level of the IoT entity under development.

Keywords. Trust, Internet of Things (IoT), Analytic Hierarchy Process (AHP), Multi Criteria Decision Analysis (MCDA), Requirements Engineering

1 Introduction

Decision making is a hard task that can become even more difficult if there are numerous, heterogeneous and conflictual aspects to be considered at the same time.

To solve this issue, Thomas L. Saaty developed the Analytic Hierarchy Process (AHP) [19]. AHP is one of the most used methodologies [22] in the Multi Criteria Decision Analysis (MCDA), a discipline that supports the decision-maker (DM) in choosing among numerous and conflicting alternatives. This methodology allows the DM to compare different alternatives in order to decide which one

better fulfils a specific goal considering both qualitative and quantitative criteria. At the end of the process, this methodology returns a global value for each criterion and alternative. According to this value, the DM is able to choose the best alternative among the others. Anyway, AHP presents some issues. In fact, it is possible to have inconsistent choices. The risk increases when the alternatives and the criteria grow.

For this reason, we propose an AHP-like methodology called Pairwise Ordination Method (POM), in order to avoid this issue. Moreover, we have decided to apply POM to the IoT field because, due to the uncertainty and the heterogeneity of this topic, we need to decide among different alternatives belonging to various fields. AHP was developed to permit pairwise comparison, taking into account that the human brain has a limited capacity to make decisions [3] if it has to choose among many alternatives and POM guarantees also this aspect.

Decision making is an important task during the System Development Life Cycle (SDLC), especially in the earliest phases. Ferraris et al. [7] proposed a framework for the Internet of Things (IoT) in order to develop an IoT entity. In this framework, it is fundamental to consider trust holistically in the whole SDLC.

Trust is a difficult concept to be defined because it is strongly connected to its context and it can differ according to the topic in which is considered [4].

Moreover, according to Pavlidis [17] and Hoffman et al. [10] trust is strongly related to other properties as privacy, identity and security. As stated by Ferraris et al. [6], it is possible to have a different type of requirements belonging to different properties. Moreover, in an environment like the IoT, we must take into consideration the context also during the requirements phase.

During the System Development Life Cycle (SDLC) of an Internet of Things (IoT) entity, it is possible to encounter conflicts among needs, requirements or choices. Especially during the requirements elicitation process, POM can be very helpful in order to decide which requirement improves the level of trust in the developing entity.

In this paper, we propose an AHP-like methodology called POM in order to solve possible arising conflicts among requirements. This approach helps the developers in deciding which requirement can guarantee the highest level of trust for the developing system ranking it among the other conflicting requirements.

The paper is structured as follows. Section 2 describes the related work and Section 3 present the background related to the K-Model and AHP. In Section 4, we explain POM methodology, while in Section 5, we describe an IoT use case scenario to illustrate the methodology. In Section 6 we discuss the results. Finally, in Section 7 we conclude the paper and discuss future work.

2 Related Work

The IoT is a network of interconnected objects. Roman [18] states that the goal of the IoT is to enable things to be connected anytime, anyplace, with anything and anyone, ideally using any path network and any service. It is expected that

these objects will often have to interact with each other often under uncertain conditions.

Mechanisms to solve this lack of information are needed and trust can help address this need [24] overcoming uncertainty [5].

Trust is a difficult concept to define "because it is a multidimensional, multidisciplinary and multifaced concept" [24].

Jøsang [11] defines trust as personal and subjective, for McKnight [14] trusting someone means to depend on him, no matter the consequences. Gambetta [9] recognises trust and the symmetrical distrust as a subjective probability to perform an action. Agudo et. al [1] define trust as "the level of confidence that an entity participating in a network system places on another entity of the same system for performing a given task".

Related to trust, reputation is defined as an objective concept. Jøsang [11] asserts that it is possible to trust someone based on his good reputation or despite his bad reputation, this means that reputation is a factor of trust but it is not the only one.

As Yan et al. [24] state "Trust management plays an important role in IoT for reliable data fusion and mining, qualified services with context-awareness, and enhanced user privacy and information security. It helps people overcome perceptions of uncertainty and risk and engages in user acceptance and consumption on IoT services and applications".

According to Hoffman et al. [10] and Pavlidis [17] trust is strongly dependent on other security properties like privacy, identity and reliability. From these definitions, Ferraris et al. [7] stated that in an IoT entity development it is important to centrally consider trust and related properties as privacy or identity in the whole system life cycle. A fundamental part is related to the requirements analysis phase and the traceability among requirements. AHP-like methodologies can improve this work in order to choose the most important requirement among conflicting requirements, to maximise the trust level of the system.

3 Background

3.1 K-Model

In a previous work [7], we have proposed a framework to guarantee trust in a smart entity during the whole system life cycle. This framework is composed of a K-Model and transversal activities (i.e. Decision Making, Traceability) as shown in Figure 1. Furthermore, the context layer is always present in each phase. In an IoT environment due to its dynamicity and heterogeneity, we always need to take context into consideration. The context can depend on the environment or on the rules of the company that develops the product.

In the K-Model, we have highlighted different phases to cover all the SDLC of a product, from cradle to grave. The first phase concerns the need phase, where it is decided the purpose of the new IoT entity and the stakeholders have a key role in it. After this phase, we define the requirements phase where the developers elicit the requirements from the previous needs.

In a requirement elicitation process it is possible to elicit requirements that are conflicting among them, this can be due to the needs arisen from different stakeholders. In this paper, we assume that the requirements elicitation process has been completed and we have to deal only with conflictual requirements. Therefore, decision-making can help in solving this issue. Useful parameters to be taken into consideration are context, traceability and requirements domains as we will discuss later.

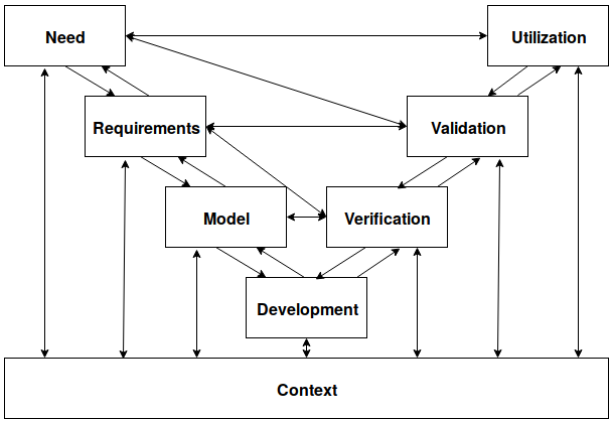

Fig. 1. K-Model: with the *Transversal Activities* it compounds the framework [7]

3.2 Analytic Hierarchy Process (AHP)

The Analytic Hierarchy Process (AHP) has been developed by Saaty [19] and it is a structured technique for organising and analysing complex decisions. It is one of the most widely used methodologies in MCDA [21].

A peculiarity of this methodology is that it is possible to compare aspects related to different fields because they are prioritised using a normalised value. This is very important because, in accordance with the multidisciplinary aspect of trust and the heterogeneity of IoT, it is useful to be able to compare alternatives from different areas and see which could be the most important with respect to a particular goal.

Basically, AHP permits to decide between various alternatives which one satisfies a predetermined goal. The alternatives have to be compared with respect to determined criteria. We can see the general AHP model in Fig. 2.

The goal that we are pursuing is located at the top layer. Then, at the intermediate layer we can see the criteria. Finally, the alternatives are located at the bottom layer.

The nodes at each layer are pairwise compared with respect to their contribution to the nodes above them. The results of these comparisons are entered

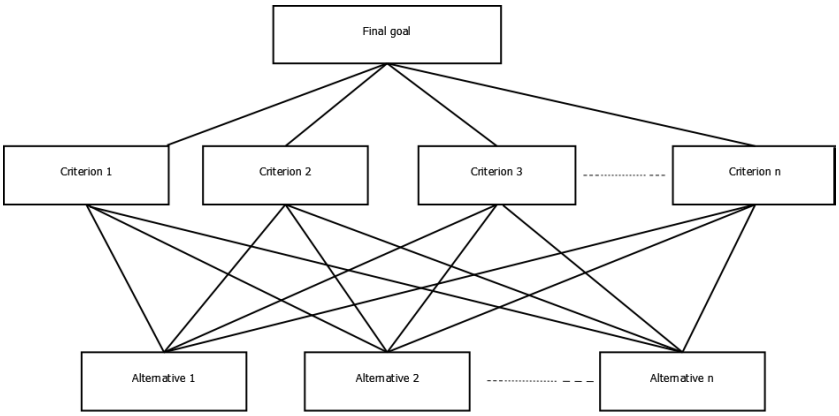

Fig. 2. General AHP model

into a matrix which is then processed mathematically to derive the priorities for all the nodes on that level. In case it is needed a higher level of detail, it is possible to divide a criterion into sub-criteria.

During the pairwise comparisons, we have to decide which term of comparison is more important, if the first one (A) or the second (B). Saaty [19] proposed a fundamental scale to compare the elements. It is shown and explained in Table 1.

Table 1. The fundamental scale for pairwise comparisons [19]

Intensity	Meaning
1	A and B are equally important
3	A is relatively more important than B
5	A is more important than B
7	A is much more important than B
9	A is absolutely more important than B

When we assign a number in a relationship according to the fact that the relationship is two-way, we have to fill the other cell with a reciprocity number. So the more important A is for B, the less important B is for A. For example, if we have to compare A and B in accordance with a criterion C and we think that A is more important than B, we give to A a value of 5 with respect to B, so we have to give to B value of 1/5 with respect to A.

The comparison usually starts comparing the criteria respect to the goal, the value of the comparison fill the matrix related to the criteria respect to the goal. After that, the DM must compare the sub-criteria with the originating criterion and the values fill the related matrix. Finally, the DM must compare

the alternatives with respect to each criterion, again the values fill the related matrix.

For every matrix, a Consistency Index (CI) is calculated. It is used to check if the choices made during the comparisons are consistent or not. If CI is lower than 0.10 [19], so the matrix is consistent. If not, the DM must reconsider the choices made before and solve the consistency issue. The difficulty grows with the number of elements and it is harder to avoid inconsistency.

AHP utilization is discouraged for more than ten alternatives or criteria [19]. This because of the risk of computing inconsistent values during the pairwise comparison. This is an issue that in an environment as the IoT can limit the effectiveness of this approach. We mitigate this issue with our methodology as we show in the next section.

AHP in the State of the Art AHP has been used in the state of the art about trust management only in a few works. One of them is [16] where they used AHP in order to address the fact that IoT nodes have limited computational power. They used the model in order to calculate the trust level using also a reputation parameter. They do not consider related properties as privacy or security. This is an interesting work that will be taken into consideration in the following phases of our work especially during the utilization phase of the K-Model.

Furthermore, AHP is applied in individual and group decision making and it is an effective and flexible tool for structuring and solving complex group decision-making situations as Altuzarra et al. stated in their work [2].

Taha et al. [21] introduced an AHP-based technique that allows a comparative analysis of cloud security. This is interesting for our perspective because their technique simplifies security requirements specifications. They consider a methodology that helps users to better understand and identify their security needs. However, in our framework [6], we have considered security requirements according to other types of requirements (i.e. availability, privacy) that in their work are missing.

In another field, Kim [13] have presented an AHP method based on network interfaces and a channel selection algorithm for multichannel MAC protocols in IoT ecosystems that considers several decision-making factors such as expected channel condition, latency and frame reception ratio. The proposed scheme considers an IoT-based healthcare system and can fit in a more complex use case scenario similar to the one that we will propose later in this paper.

Finally, Kassab et al. [12] explore AHP in order to assist the prioritization of quality requirements. This approach can be limited due to the increase of complexity if it is used for all the requirements. In our approach, we focus only on the conflictual requirements.

4 Methodology: Pairwise Ordination Method (POM)

Our approach is based on AHP. It is composed of a goal, alternatives and criteria as it is for AHP, but the operation among these elements are performed differ-

ently. The goal is to rank the requirements that guarantee the maximum level of trust for the developing IoT entity according to the criteria and the conflicting alternatives.

The criteria, as long for AHP, can be divided into sub-criteria to improve the level of detail.

Finally, there are the alternatives, that in our case are the conflicting requirements.

In this approach, trust is considered as a way to improve the quality of the system in choosing which requirement to keep and which one to release.

4.1 Trust

Trust can comprise multiple aspects. It can be strictly dependent on the context, on the time or on the reliability of a particular choice.

Depending on the goal and the chosen alternatives, it is possible to decide which trust-related criteria are most important in order to fulfill the final goal. The criteria can be general or specific.

A peculiarity of this methodology is that is possible to compare aspects related to different fields prioritising them with a normalised value. Thus, we can compare criteria such as time with others like cost choosing between them. This aspect is very important because, in accordance with the multidisciplinary aspect of trust, it is useful to be able to compare aspects belonging to different areas.

The goal we have to achieve is to choose the requirement maximising the trust level of the developed system. To fulfil this trusted goal, we have identified three groups of criteria that are mandatory to use in our methodology:

1. **Context criteria.** As we stated in [7], the context can be a composition of functionalities or depending on the environment. Context is always present and needed to be taken into consideration in an IoT environment. Moreover, these criteria can affect the trust value of the whole system. Furthermore, according to the general aspects related to trust, we have to identify *general criteria* that can affect the trust level of the system.
2. **Traceability criteria.** As we stated in [7], traceability is crucial during the development of an IoT entity. Moreover, we cannot solve a conflict requirements issue without considering which other requirements, needs, models or documents are involved and connected with the conflicting requirements. For this reason, the complexity of the IoT entity under development must be taken into consideration in order to decide how to proceed. Also, validation and verification aspects must be taken into consideration in this group criteria. When we write a requirement, we must decide how to verify and validate it. For this reason, if we modify such requirement, we have to modify as well the way we can verify and validate it.
3. **TrUStAPIS criteria.** These criteria are related to the characteristics of trust highlighted in [6]. In addition, as we stated also in [7], trust is strongly related to other properties: privacy, identity, security, usability, safety and

availability. The domains of these criteria are the same as the conflicting requirements. It is useful to perform this comparison because it is important to rank them accordingly to other connected requirements in order to see which criterion is more general and useful compared to all the requirements. It can be only one (in the case all the conflicting requirements belong to the same domain) or there can be seven criteria (all the domains identified in [6]).

We will show how these criteria can be covered in the use case scenario proposed in Section 5.

The alternatives that we take into consideration in this method are requirements that arise conflicts among them. Furthermore, we assume that the elicitation process and the identification of these conflicts have been already done. Depending on the alternatives, it is important to define the criteria that permit to decide between them. The criteria must belong to the classification groups we have shown before.

4.2 Procedure

In this methodology, we have to perform comparisons between the elements belonging to contiguous layers. We start comparing goals and criteria. Then, we compare criteria and sub-criteria (if they exist) and finally, we compare the alternatives with criteria or sub-criteria.

These comparisons are necessary to order alternatives and criteria from the less to the more important. At the end of the procedure, we can rank the requirements from the most to the least important. The importance is given by which one of the requirements is most important in order to improve the level of trust of the system.

During each comparison, we create an ordered branch-tree. For each round of comparison, we compare an element with the following not yet ordered elements and the DM decides which one is more important in accordance with the criterion or goal. The procedure is iterative and each time an element *wins* the comparison, it has to be compared with the following elements. The round ends after each element has been compared. When the first round finishes, we obtain a partially ordered tree. In the following rounds of comparisons, we will compare the not yet ordered elements until we have a completely ordered branch-tree. Anyhow, it is possible that this branch should have some levels populated by more elements.

For example, let us assume that we have six alternatives (A,B,C,D,E and F) and we have to compare them according to criterion X.

In the first round, we have to compare all the elements among them in order to decide which one is the most important. Thus, firstly, we compare A with B in order to decide which alternative is more important. Imagine that the DM decides that B is the most important. It means that B will be then compared with C. Let us assume that B wins all the comparisons until F. So, we find out that B is the most important alternative among the others according to the criterion X, but we do not know the rank of the other alternative. So, we have to

estimate also the order of the other alternatives and we start the second round of comparisons.

Firstly, we compare A and C and we find out that C is most important. Secondly, C is compared with D and the DM decides that D is more important than C. Then, we have to compare D with E and D wins. Finally, we compare D with F and D is the most important.

Now, we know that B is the most important and the second element is D. But we have to order also the remaining alternatives.

In the third round, we compare A and C again, but we previously found out that C was more important than A, so it is possible to skip this comparison. Then, we compare C and E deciding that C is the most important. Finally, C is compared with F and C wins again.

Now, our branch-tree is composed of B, C and D ranked in this order. The remaining alternatives to be ordered are A, E and F.

Thus, we start the fourth round comparing A and E and we find out that they are equally important according to X. Finally, we compare A with F and the DM decides that A is more important.

In the last round of comparisons, we find out that E is more important than F because A and E have the same importance.

So, the algorithm ends and we have an ordered branch-tree that is shown in Figure 3.

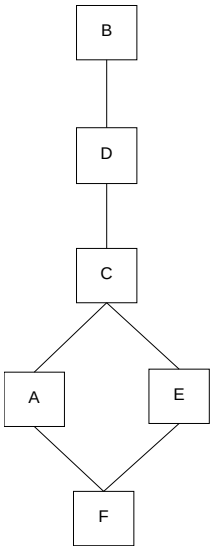

Fig. 3. Example: ordered branch-tree

Now that the branch-tree is ordered, we need to give each element weight to be normalized and compared with the other branch-trees related to other

criteria. In our approach, the weights are based on the number of elements and on the order that they have. The maximum weight value is equal to the number of elements, the minimum value is one. In order to assign the weights, we proceed following a bottom-up approach. If the bottom layer has only one element, the element belonging to the layer above will have a value equal to the lower value plus one. In the case a layer has more than one element of the same importance they will have the same value. Anyway, in this case, the element above them will have a value equal to their value plus the number of equal elements. We use this "jump" to highlight the difference between the upper element respect to the lower elements.

We can summarize these cases in the following general formula:

$$Element\_weight = Lower\_element\_weight + Number\_of\_lower\_elements$$

We can consider the previous example in order to show how the weights are given.

We had six elements (A,B,C,D,E and F). So, we have 6 as the maximum value and 1 as the minimum value. So, we give to F weight 1 because it is the lowest. Then, we have two elements of the same importance (A and E) and we give to both of them the weight 2. Above them, there is C that has a weight of 2 (the value of the element beneath it) plus 2 (the number of elements of the same value in the lower level). So, the weight of C is 4. Then, at the level above, we have D that is equal to 5. Finally, we have the most important element that has a weight of 6. Our weighted and ordered branch-tree is shown in Figure 4.

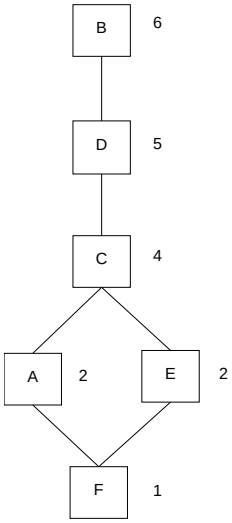

Fig. 4. Example: ordered and weighted branch-tree

Considering that in a real example there will be other criteria, we have to normalize these values in order to compare them with the results related to the other criteria.

Finally, we have to compare the criteria among them according to the final goal in order to rank them.

Considering the previous example, to normalize the elements we have to divide each of them for the sum of the weights that is 20. Thus, the normalized values for each alternative will be:

$B = 6/20 => 3/10$
$D = 5/20 => 1/4$
$C = 4/20 => 1/5$
$A = 2/20 => 1/10$
$E = 2/20 => 1/10$
$F = 1/20$

The sum of all the weights is 1. In order to reach the final goal and to choose which alternative is most important. Each alternative will be multiplied for the weight of the sub-criteria and criteria among them and added to the values of the same alternative compared to the other sub-criteria and criteria.

The final sum of all these values will be 1 and the higher alternative will be the most important.

We will show this procedure with a complete example in Section 5.

4.3 POM vs AHP

The complexity of POM is the following: the maximum number of comparisons depends on the number of alternatives and the minimum number of comparisons is the number of the alternatives minus one, as we show in Table 2. The latter is the best case, in fact, imagine that with each comparison the following element always wins, it means that the previous elements have been already ordered and we need only one round of comparisons to order the branch-tree.

Table 2. Maximum comparison related to the number of alternatives. Legenda: i = number of alternatives , max = maximal number of comparisons, min = minimum number of comparisons

i	1	2	3	4	5	6	7	n
max	n.a.	1	3	6	10	15	21	$n(n-1)/2$
min	n.a.	1	2	3	4	5	6	$n-1$

In AHP, we have a fixed number of comparisons that is equal to the maximum number of comparisons of our method:

$$AHP - comparisons = n(n-1)/2$$

An issue that is possible to face using AHP is the increasing possibility to have an inconsistent matrix when the number of elements to be compared grows. With this methodology, the inconsistency is avoided because the comparison tree is ordered at every step of the algorithm.

Another difference between AHP and our methodology is related to the weights. Every element must be compared with the other elements according to the goal (for the criteria), to the criterion (according to the sub-criteria or the alternatives) and to the sub-criteria or the criterion (according to the alternatives). If in AHP, the weights are shown in Table 1, in our approach the weights are dependent on the number of elements as we have shown earlier.

Summarizing, in our methodology the weights are fixed. On the other hand, using AHP the weights can represent better the differences among the elements. Anyhow, this possibility can increase errors due to subjective decisions. This error is mitigated by our approach.

5 Use Case Scenario: Smart Hearth-monitor

We propose a use case concerning a health service scenario. In this scenario, we assume that during the development of a new heart monitor an issue has arisen. The issue is due to three different needs belonging to different stakeholders creating conflict among them. The three stakeholders are the vendors and customers (patients and doctors). We refer to the elicited requirements following the TrUStAPIS methodology [6].

For the vendors, it is necessary to know the more data is possible about the customers for market surveys. For the customer is necessary to remain anonymous or at least they need to know that only trusted users can access their data. For the doctors is necessary to know at least age, gender, diseases and blood type of the user in order to be helpful for the user. The patients want to be monitored but they care about privacy and trust in order to submit their personal data. The vendors want to monitor and improve the product in order to sell it. For this reason, they need a huge amount of data from the customers (i.e. age, gender, nationality, address).

We have identified five conflictual requirements and, without solving this issue, it is not possible to continue the development process of the product. The requirements are one privacy, one trust, one identity and two availability requirements. They are better explained in Section 5.2.

The criteria belong to the ones identified in Section 4.1 and are defined according to this scenario in Section 5.1.

The goal is to decide which requirement to keep that can maximize the level of trust of the IoT entity perceived by the "conflicting" stakeholders.

5.1 Criteria

According to our scenario, we have considered the following important aspects as decision criteria:

– **Context criteria**

1. **Stakeholders Importance.** When a product is developed, there are different stakeholders that can take decisions and they can have different importance. In fact, a vendor could stop the project or the customers could not buy the product. For example, if a vendor decides to stop the project the customers will never have it, on the other hand, if the customers will not buy the product, it will be a failure. In order to be more specific, this criterion needs to be divided into three sub-criteria.

   (a) *Vendors*. For the vendors, it is important to have as much information as possible about their customers.

   (b) *Doctors*. The doctors need to know all the useful information about the patients to be able to treat their diseases (i.e. blood type, historical diseases).

   (c) *Patients*. Patients do not want to share their sensitive information with untrusted users. They can accept at least to share the minimum information as possible with trusted users in order to be properly cured.

2. **Faster.** It is possible that a requirement could be easier than another in order to be implemented. If a requirement is very hard to be achieved, it can be relaxed in case of conflicts with another. In case of a strict deadline, this can be the most important parameter. It can be an objective parameter.

3. **Cheaper.** This criterion is about the cost of the implementation. In case of conflicting requirements and a limited budget, this can be the most important parameter to be taken into consideration. It is an objective parameter.

– **Traceability criteria** These criteria are objective. In fact, according to these criteria, the more connections with other elements are developed, the more important the requirement is.

1. **Connected Requirements.** This knowledge is guaranteed by the traceability database [6]. Thus, if a requirement is released, then the connected requirements can be affected by that. This is an objective parameter. The more requirements are connected, the more important the requirement is.

2. **Connected Needs.** Requirements derive from needs, so this is an important element to be taken into consideration. This knowledge is guaranteed by the documentation activity [7]. Thus, if a requirement is released or must be changed, it is possible to go back to the originating

need and change it in accordance with the stakeholders. This is an objective parameter and its importance depends on the connected needs.

- **TrUStAPIS criteria** These criteria depend on the type of requirements chosen as alternatives. It is important in order to highlight how much a requirement belonging to a particular domain (i.e. privacy) is connected to another domain (i.e. trust). In our use case, the domains are the following: trust, security, privacy and availability.

We need to compare the main criteria according to the goal and the sub-criteria according to the originating criterion. We show the process in Section 6.

5.2 Alternatives

The alternatives are one trust requirement, one privacy requirement, one identity requirement and two availability requirements.

1. **Privacy Requirement.** PRIV01 - The customer shall remain anonymous.
2. **Availability Requirement.** AVBT01 - The doctor shall access patients information related to his/her pathology.
3. **Availability Requirement.** AVBT02 - The vendor shall access to all customer information.
4. **Identity Requirement.** IDNT01 - The customer shall provide his/her personal data to the system.
5. **Trust Requirement.** TRST01 - The customer data shall be accessed only by trusted users.

We need to compare each one to the others according to the criteria and sub-criteria we have identified before.

These requirements create conflict among them because they are belonging to different stakeholders with incompatible needs. Our methodology helps to choose the most important requirement in order to assure the highest possible trust value on the IoT entity. In some cases, changing the requirements means that it is needed to change also the originating need.

As we explained earlier, the alternatives must fulfil the goal. In this scenario, it is important to rank the requirements in order to release or change the less important requirements and keep the others avoiding conflicts.

Now that we have presented criteria and alternatives we can build the POM model according to the use case scenario proposed. It is shown in Fig. 5.

For the sake of simplicity, we represent the connection among the alternatives and the criteria with a single point of contact. The dotted line represents a many-to-many connection among them.

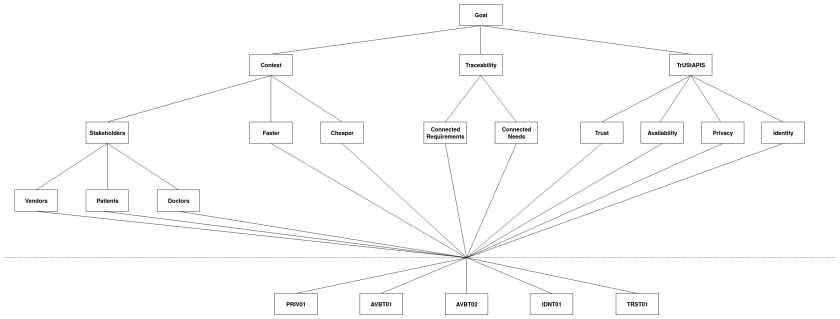

Fig. 5. POM model related to our use case scenario

6 Results and discussion

We start the process by comparing the criteria according to the final goal. Secondly, we will compare the sub-criteria to their main criterion. Finally, we will compare the alternatives to their main (sub)criterion. We will move following the ordered tree shown in Figure 5. Each section is named as the element considered for the comparisons. We will not represent all the rounds of comparisons in all of the following sections for the sake of simplicity and space limitation, but we have used the methodology as we have shown earlier.

6.1 Goal

We can state that the goal is reached considering the following elements:

$$Goal = \{Context, Traceability, TrUStAPIS\}$$

We need to create a ranked order among criteria to show which of them is most important according to the goal. So, following our methodology, we start comparing the Context (Cx) with the Traceability (Ty). We decide that Cx is most important, so then we have to compare Cx with TrUStAPIS (Ts) and we decide that Cx is more important than Ts. For both the decisions, we consider the context as crucial because as presented in [7] it is an element always present and strictly connected to a particular parameter. After the first round, we know that Cx is the most important element. We need another round in order to decide between Ty and Ts. We decide that Ty is the most important because the more requirements or needs are connected, the most important is the requirement.

Thus, we will have the following normalized values:
Ts = (1/6)
Ty = (1/3)
Cx = (1/2).

6.2 Context

The context is composed of three sub-criteria: the stakeholders (Sk) criterion and others strictly dependents on the requirements, faster (Fs) and cheaper (Ch). We need to compare them in order to decide which one is the most important. Comparing the stakeholders to the faster and cheaper criteria, we decide that the stakeholders are the most important because they are the ones who provide the needs of the product. Then, we give to faster and cheaper criteria the same importance.

Thus, the values related to the sub-criteria of the context are:
Sk = (3/5)
Fs = (1/5)
Ch = (1/5).

6.3 Traceability

Traceability is a transversal activity of the K-Model [7] and it is very important for the requirements phase connecting requirements among them and also connecting different phases. In this case, there are two sub-criteria. They are related to the connected requirements (Con_R) and the connected needs (Con_N). We decide to give more importance to Con_N because the needs are the real motivation behind a product, so, if a requirement is strictly connected to a need it should be more important than another only connected to a requirement.

The values related to them are:
Con_N = (2/3)
Con_R = (1/3).

6.4 TrUStAPIS

In order to decide which requirement to keep, it is important to study the relationships among the requirements domains and the conflicting requirements. This criterion gives a holistic view of how a requirement of a particular domain can be related to others.

In this case, we compared the trust (Tt) domain firstly with the availability (Av) domain and we gave more importance to trust. Secondly, to the privacy (Pr) domain and again we decide that trust is more important (according to our particular goal). Finally, we compare it to the identity (Id) domain giving more importance to trust.

During the second round, we compare availability to privacy deciding that the latter is more important. Then, we compare the privacy requirement to the identity requirement and we decide that they are equals. For this reason, there is no need of a third round of comparison (privacy is more important than availability and consequently because identity is equal to privacy, it is more important than availability, too).

The values related to the trUStAPIS sub-criteria are:
Tt = (4/9)

Id = (2/9)
Pr = (2/9)
Av = (1/9).

6.5 Stakeholders

Stakeholders are very important for any project. They are the actors having an interest in the system. We have identified three main stakeholders strictly related to the conflict requirements. They are the vendors (Vn) and the customers (divided into doctors (Dr) and patients (Pt). We decided that they are equally important, in fact, it is true that if the vendors do not produce the product it will not be used by the users, but it is also true that if the customers will not trust and use it, the product will be a failure.

For this reason, the values are the same for all the stakeholders:
Vn = (1/3)
Dr = (1/3)
Pt = (1/3).

Now that we have ranked all the (sub)criteria, we have to rank the alternatives showing their importance about the (sub)criteria.

6.6 Vendors

Starting from the left of Figure 5, we see that there is the first sub-criterion of the stakeholders criterion. The vendors are the producers of the IoT device and the requirements are ordered considering how important are for them.

The best way to make this order is by asking them directly which requirement they prefer, but in this use case, we assume that the developer can perform this task analyzing the collected needs and requirements.

For the vendors, after the first round of comparison, we find out that the AVBT02 is the most important requirement. Secondly, IDNT01. Thirdly, TRST01 followed by AVBT01 and finally PRIV01. This order is due because AVBT02 is the only requirement that considers directly the vendors and it is the one that represents more their interests on the IoT entity. Then, in order to have the needed information, IDNT01 satisfies their need. Thirdly, TRST01 can be important for them to avoid that this information could be manipulated by malicious entities. AVBT01 is ranked as fourth because at least some information can be available with this requirement. The last one is PRIV01, because if the patients and doctors remain anonymous, the vendors will not have valuable information for market purposes.

The normalized values are the following:
AVBT02 = (1/3)
IDNT01 = (4/15)
TRST01 = (1/5)
AVBT01 = (2/15)
PRIV01 = (1/15).

6.7 Patients

For the patients, the most important requirement is PRIV01, because it is the one who guarantees them to be anonymous. Secondly, TRST01 can be a good compromise, because at least they know that only a trusted user can manipulate their information. The third one is AVBT01, because in the case anonymity is not possible, they accept that the doctors can access their pathology information in order to be properly cured. Then, IDNT01 can be accepted if the previous requirements are kept. Finally, AVBT02 is the last choice because they do not want the vendors to access to all their information.

Thus, the rounds of comparisons produce these normalized values:
PRIV01 = (1/3)
TRST01 = (4/15)
AVBT01 = (1/5)
IDNT01 = (2/15)
AVBT02 = (1/15).

6.8 Doctors

For the doctors, it is very important to have patients information related to their pathology in order to cure them efficiently and effectively. For this reason, the most important requirement for them is AVBT01. Then, TRST01 is important because also their data must be inserted in the system and they want that only trusted users can access them. IDNT01 is the third one because it is important in order to have patients data. The fourth one is PRIV01 and the fifth is AVBT02. Both of them are not important for them, but they at least prefer PRIV01 because they can be anonymous in the system and the diseases of the patients must be revealed respecting their privacy.

According to the doctors, the final values are:
AVBT01 = (1/3)
TRST01 = (4/15)
IDNT01 = (1/5)
PRIV01 = (2/15)
AVBT02 = (1/15).

6.9 Faster

This criterion and the following are strictly dependent on the context and on the developers' skills. So, they need to decide how the requirements must be ordered.

After the first round of comparisons, we have identified two equally important requirements: AVBT02 and IDNT01. In fact, these two requirements do not require any filters. They are simply the operation of reading and writing data in the database. Then, AVBT01 is more difficult to implement because it requires a specific filter. The fourth one is TRST01 because it needs a trust decision model or an authentication process in order to be implemented. Finally, the slower

requirement to be implemented is PRIV01 because it requires to anonymize the data.

According to the faster sub-criterion, the final values are:
AVBT02 = (2/7)
IDNT01 = (2/7)
AVBT01 = (3/14)
TRST01 = (1/7)
PRIV01 = (1/14).

6.10 Cheaper

This criterion is very important in the case the budget is limited or the stakeholders want to maximise the income. On the other hand, using only this criterion can be dangerous because removing a requirement in the first phases of the SDLC only because it is expensive can create issues in the following phases of the SDLC. In this case, the amount of money spent to solve the issues arisen will be higher [8].

However, after the first round of comparison, IDNT01 is considered the cheapest requirement to be developed. Then, AVBT02 and TRST01 are considered equally. Finally, there are AVBT01 followed by PRIV01. The last one is both the slower and the more expensive because to implement the anonymity is the hardest task considering the other requirements. IDNT01 is considered the cheapest because it is the basic requirement to implement.

The normalized values are:
IDNT01 = (5/14)
TRST01 = (3/14)
AVBT02 = (3/14)
AVBT01 = (1/7)
PRIV01 = (1/14).

6.11 Connected Requirements

With a more complex or real use case scenario this criterion can be completely objective, the operation needed is to count how many requirements are connected to the requirement under consideration. In this use case, we have to subjective consider the possible relationships among the requirements.

Thus, because we consider trust as central and more connected to the other domains, we consider it as the most connected requirement. Then, we consider that AVBT01 should be the second one because it refers also to the patients and to a particular type of data. Then, we consider AVBT02 and IDNT01 equally important. Finally, there is PRIV01. We assume that even if it is the harder requirement to be achieved it does not need many other requirements to be connected with in order to be elicited and developed.

Thus, the final normalized values about Con_R are:
TRST01 = (5/14)

AVBT01 = (2/7)
AVBT02 = (1/7)
IDNT01 = (1/7)
PRIV01 = (1/14).

6.12 Connected Needs

In this case, it is important to consider how many needs are connected to a single requirement. It is possible that more needs originate a single requirement or even that a requirement is not connected to any need but only to other requirements (especially if it is a sub-requirement [6]).

In the case of a real use case scenario, it is possible to count all the connected needs and apply our methodology in an objective way. However, in this case, we proceed as for the previous element.

After the first round of comparisons, we decide that IDNT01 is the requirement connected to more needs. In fact, only after the operation explained in the requirement, it is possible to process the data. For this reason, many needs shall be connected to it. The second requirement is PRIV01. In fact, we assume that not only the customers require it but also data protection regulations needs(i.e. GDPR [23]) are connected to the privacy requirement. The third "winning" requirement is TRST01. Then AVBT02 has more connected needs than AVBT01, because we imagine that the vendors want to satisfy more needs than the doctors obtaining all the customer information.

The final values are:
IDNT01 = (1/3)
PRIV01 = (4/15)
TRST01 = (1/5)
AVBT02 = (2/15)
AVBT01 = (1/15).

6.13 Trust

This criterion is related to trust and we want to prioritize all the requirements according to this domain. This operation is interesting not for the first requirement (that with high probability will belong to the same domain), but for the following, in order to decide which of them is more connected to trust (in this case).

Thus, the most important requirement is TRST01. Then, we have IDNT01, because, in order to perform this operation, the customers must trust the system, otherwise they will not use it. Then, AVBT02 is another requirement that implies a high level of trust in the system. Because, only if the customers can trust the vendors, it should be possible to implement it. Then, there is AVBT01 that is related to the trust between patients and doctors. Finally, there is PRIV01 that must be implemented if there is a low level of trust between customers and vendors.

The final normalized values are:
TRST01 = (1/3)
IDNT01 = (4/15)
AVBT02 = (1/5)
AVBT01 = (2/15)
PRIV01 = (1/15).

6.14 Availability

The two availability requirements are the most important according to this criterion. Then, IDNT01 is the third one, because only if the customers provide their information, they will be available. Then, TRST01 is important for availability, because it guarantees that the data are available for trusted users. Finally, PRIV01 is the least important requirement according to availability, because if the data are anonymous, they are more difficultly available.

For availability, the normalized values are:
AVBT01 = (2/7)
AVBT02 = (2/7)
IDNT01 = (3/14)
TRST01 = (1/7)
PRIV01 = (1/14).

6.15 Privacy

According to privacy, the most important requirement is PRIV01. Then, the second one is AVBT01, because it guarantees that only the doctors can access to sensitive information. TRST01 is the third parameter because it guarantees that only trusted users can access to the data. Then, there are both AVBT02 and IDNT01.

The final values are:
PRIV01 = (5/14)
AVBT01 = (2/7)
TRST01 = (3/14)
IDNT01 = (1/14)
AVBT02 = (1/14).

6.16 Identity

For the identity domain, the most important requirement is IDNT01. Then, there is AVBT02 because it requires that the identity data should be available for the vendors. The third requirement is AVBT01 because it provides at least some data for the doctors and they need to know the identity of the patients in addition to their diseases. Then, there is TRST01 and finally PRIV01. The latter does not provide any identity information.

The final normalized value about identity are:
IDNT01 = (1/3)

AVBT02 = (4/15)
AVBT01 = (1/5)
TRST01 = (2/15)
PRIV01 = (1/15).

6.17 Final priority

After the calculation of the normalized local weights, we have to compute the final priority related to each of the alternatives according to the goal. In order to do this, we must sum every value related to the single alternatives multiplying it for each value of the sub-criteria and criteria above it.

Because the values are normalized, the final results and their sum will be 1.

In Figure 6 are shown the values derived from section **??** to **??**.

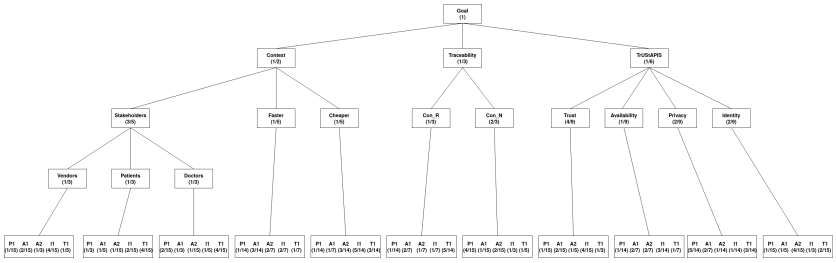

Fig. 6. POM model related to our use case scenario

In this figure, to avoid having many lines and boxes for the alternatives, we have summarized them in a single box. P1 is the privacy requirement, A1 and A2 are the availability requirements, I1 is the identity requirement and T1 is the trust requirement.

Thus, in order to calculate the single priority related to the conflicting requirements, starting from PRIV01, we will have.

$$\mathbf{P1} = (1/15) * (1/3) * (3/5) * (1/2) + (1/3) * (1/3) * (3/5) * (1/2) + (2/15) * (1/3) * (3/5) * (1/2) + (1/14) * (1/5) * (1/2) + (1/14) * (1/5) * (1/2) + (1/14) * (1/3) * (1/3) + (4/15) * (2/3) * (1/3) + (1/15) * (4/9) * (1/6) + (1/14) * (1/9) * (1/6) + (5/14) * (2/9) * (1/6) + (1/15) * (2/9) * (1/6) = 0.157$$

For the other requirements, we have:

$$\mathbf{A1} = (2/15) * (1/3) * (3/5) * (1/2) + (1/5) * (1/3) * (3/5) * (1/2) + (1/3) * (1/3) * (3/5) * (1/2) + (3/14) * (1/5) * (1/2) + (1/7) * (1/5) * (1/2) + (2/7) * (1/3) * (1/3) + (1/15) * (2/3) * (1/3) + (2/15) * (4/9) * (1/6) + (2/7) * (1/9) * (1/6) + (2/7) * (2/9) * (1/6) + (1/5) * (2/9) * (1/6) = 0.182$$

$$\mathbf{A2} = (1/3) * (1/3) * (3/5) * (1/2) + (1/15) * (1/3) * (3/5) * (1/2) + (1/15) * (1/3) * (3/5) * (1/2) + (2/7) * (1/5) * (1/2) + (3/14) * (1/5) * (1/2) + (1/7) * (1/3) * (1/3) + (2/15) * (2/3) * (1/3) + (1/5) * (4/9) * (1/6) + (2/7) * (1/9) * (1/6) + (1/14) * (2/9) * (1/6) + (4/15) * (2/9) * (1/6) = 0.175$$

$$\mathbf{I1} = (4/15) * (1/3) * (3/5) * (1/2) + (2/15) * (1/3) * (3/5) * (1/2) + (1/5) * (1/3) * (3/5) * (1/2) + (2/7) * (1/5) * (1/2) + (5/14) * (1/5) * (1/2) + (1/7) * (1/3) * (1/3) + (1/3) * (2/3) * (1/3) + (4/15) * (4/9) * (1/6) + (3/14) * (1/9) * (1/6) + (1/14) * (2/9) * (1/6) + (1/3) * (2/9) * (1/6) = 0.253$$

$$\mathbf{T1} = (3/15) * (1/3) * (3/5) * (1/2) + (4/15) * (1/3) * (3/5) * (1/2) + (4/15) * (1/3) * (3/5) * (1/2) + (1/7) * (1/5) * (1/2) + (3/14) * (1/5) * (1/2) + (5/14) * (1/3) * (1/3) + (1/5) * (2/3) * (1/3) + (1/3) * (4/9) * (1/6) + (1/7) * (1/9) * (1/6) + (3/14) * (2/9) * (1/6) + (2/15) * (2/9) * (1/6) = 0.233$$

The results are rounded to three digits after zero. The total is:

$$P1 + A1 + A2 + I1 + T1 = 1$$

We have found out that the most important requirement is IDNT1 followed by TRST01. On the other hand, the least important requirement is PRIV1. For this reason, it will be the requirement released or changed in order to solve the conflict requirements. If this operation does not solve the conflicts, it is possible to repeat the calculations (in the case the privacy requirement has been modified) or to select the second least requirement (AVBT02) and delete or modify it. The process is iterative until the conflicts are solved.

7 Conclusion and Future Work

This example shows the potential of a process AHP-like in order to assist the decision-makers. We think that our approach can be useful in an IoT environment because of the heterogeneity and dynamicity of this field. In addition, can be very helpful in later phases of the system development life cycle, as the utilization [7]. Because of the low computational power of the smart objects, a low computation effort guaranteed by our methodology can be helpful.

Even if the subjective analysis is keen to know issues, especially if performed by different users, we mitigate this issue considering only specialized actors performing the comparisons belonging to their field (i.e a specific stakeholder or the developers). In addition, if there are objective parameters, it is possible to directly consider them in order to perform the comparisons.

Another important aspect to be taken into consideration is the increasing of the complexity when the alternatives and criteria grow, as stated also by Metcalfe [15]. This methodology limits the computational effort required and avoids the problems that AHP has when the complexity increases [19].

As future work, we will compare the results of our methodology against AHP in the same scenario. Moreover, this work will be extended to a real use case

scenario. Through this approach, comparative studies might be conducted to analyse similarities or differences among trust and reputation models. We will compare it also aganist Analytic Network Process [20] under a trust perspective.

Acknowledgement

Insert which projects to acknowledge.

References

1. Agudo, I., Fernandez-Gago, C., Lopez, J.: A model for trust metrics analysis. In: International Conference on Trust, Privacy and Security in Digital Business. pp. 28–37. Springer (2008)
2. Altuzarra, A., Moreno-Jiménez, J.M., Salvador, M.: A bayesian priorization procedure for ahp-group decision making. European Journal of Operational Research 182(1), 367–382 (2007)
3. Cabała, P.: Using the analytic hierarchy process in evaluating decision alternatives. Operations research and decisions 20(1), 5–23 (2010)
4. Erickson, J.: Trust metrics. In: Collaborative Technologies and Systems, 2009. CTS'09. International Symposium on. pp. 93–97. IEEE (2009)
5. Fernandez-Gago, C., Moyano, F., Lopez, J.: Modelling trust dynamics in the internet of things. Information Sciences 396, 72–82 (2017)
6. Ferraris, D., Fernandez-Gago, C.: Trustapis: a trust requirements elicitation method for iot. International Journal of Information Security pp. 1–17 (2019)
7. Ferraris, D., Fernandez-Gago, C., Lopez, J.: A trust by design framework for the internet of things. In: NTMS'2018 - Security Track (NTMS 2018 Security Track). Paris, France (Feb 2018)
8. Friedenthal, S., Moore, A., Steiner, R.: A practical guide to SysML: the systems modeling language. Morgan Kaufmann (2014)
9. Gambetta, D., et al.: Can we trust trust. Trust: Making and breaking cooperative relations 13, 213–237 (2000)
10. Hoffman, L.J., Lawson-Jenkins, K., Blum, J.: Trust beyond security: an expanded trust model. Communications of the ACM 49(7), 94–101 (2006)
11. Jøsang, A., Ismail, R., Boyd, C.: A survey of trust and reputation systems for online service provision. Decision support systems 43(2), 618–644 (2007)
12. Kassab, M., Kilicay-Ergin, N.: Applying analytical hierarchy process to system quality requirements prioritization. Innovations in Systems and Software Engineering 11(4), 303–312 (2015)
13. Kim, B., Kim, S.: An ahp-based interface and channel selection for multi-channel mac protocol in iot ecosystem. Wireless Personal Communications 93(1), 97–118 (2017)
14. McKnight, D.H., Chervany, N.L.: The meanings of trust (1996)
15. Metcalfe, B.: Metcalfe's law: A network becomes more valuable as it reaches more users. Infoworld 17(40), 53–53 (1995)
16. Pang, X.Y., Wang, C.: The study of trust evaluation model based on improved ahp and cloud model in iot. In: Advanced Materials Research. vol. 918, pp. 258–263. Trans Tech Publ (2014)

17. Pavlidis, M.: Designing for trust. In: CAiSE (Doctoral Consortium). pp. 3–14 (2011)
18. Roman, R., Najera, P., Lopez, J.: Securing the internet of things. Computer 44(9), 51–58 (2011)
19. Saaty, T.L.: Analytic hierarchy process. Wiley Online Library (1980)
20. Saaty, T.L.: Analytic network process. In: Encyclopedia of Operations Research and Management Science, pp. 28–35. Springer (2001)
21. Taha, A., Trapero, R., Luna, J., Suri, N.: Ahp-based quantitative approach for assessing and comparing cloud security. In: Trust, Security and Privacy in Computing and Communications (TrustCom), 2014 IEEE 13th International Conference on. pp. 284–291. IEEE (2014)
22. Vaidya, O.S., Kumar, S.: Analytic hierarchy process: An overview of applications. European Journal of operational research 169(1), 1–29 (2006)
23. Voigt, P., Von dem Bussche, A.: The eu general data protection regulation (gdpr). A Practical Guide, 1st Ed., Cham: Springer International Publishing (2017)
24. Yan, Z., Zhang, P., Vasilakos, A.V.: A survey on trust management for internet of things. Journal of network and computer applications 42, 120–134 (2014)