International Journal of Academic Research & Technology (IJART)
Volume: 01 Issue: 01

INSPECTING THE POLICY CONFLICTS IN DISTRIBUTED SYSTEM MANAGEMENT

Shreyashi Kundu, Ph.D Scholar (Commerce) St. Xavier's University, Kolkata, West Bengal, India

Abstract:

The term "distributed system management" refers to the processes that must be followed in order to ensure that big, dispersed networks will function in a manner that is consistent with the goals of their users. These goals are generally outlined in policies, which system managers can then use to guide their decision-making. There are advantages to be gained from automating management jobs that are repetitious or from giving human administrators with automated assistance. In order to accomplish this goal, it is optimal to provide a model of policies in the form of objects that the framework itself might potentially represent. In order for an automated framework to effectively recognize and handle them, it must first do an analysis of the many types of disputes that could arise. This is necessary due to the fact that human administrators might be able to address these disagreements in a civilized manner. There are not one but two different types of policies. To begin, authorization policies and obligation policies detail the responsibilities of management in terms of what they are permitted to do and what they are expected to carry out, respectively. Policies, which are themselves referred to as objects, are what determine the relationship that exists between managed subjects (managers) and managed objects (targets). Utilize domains so that you may organize the things in your system to which a policy applies. Second, obligation policies provide a framework for managerial decision-making by outlining the actions that a manager is required or forbidden to take. The manager is responsible for interpreting the commitments in order to achieve the overarching goals of the organization.

Keyword: Management policy, policy conflicts, authority, conflict resolution, distributed system management, domains.

1. Introduction:

When it comes to connecting with one another and managing their own internal activities, businesses are finding that large dispersed computing networks are becoming an increasingly necessary component. Frequently, they are composed of a number of interconnected networks that serve as the operational basis for a variety of different types of

organizations. Programs of this sort require proactive management as opposed to reactive management for a variety of reasons. To begin, there are already a number of organizations that are dependent on distributed networks to carry out their day-to-day operations. Since these networks have transitioned from supporting to organizational functions, they require a proactive plan to ensure that they are in excellent working condition. Second, the administration of the machine is typically decentralized, which means that it requires the collaboration of a large number of administrators in order to continue to function well. Third, despite their decentralized nature, distributed networks are extremely dynamic in a variety of different ways. They may be made up of hundreds of thousands of resources and be used by thousands of users; they may be dispersed across vast geographic areas, international boundaries, various regulatory bodies, and different time zones; they may contain a variety of materials that were produced by a number of different manufacturers; they may be dispersed across vast geographical areas, international boundaries, and various regulatory authorities; and they may be distributed across large geographical areas, different time zones. Components of distributed system management include the monitoring of system activity, the selection of appropriate management strategies, and the execution of control actions 1. Policy is an example of a category of information that can have an effect on the behaviour of system objects. The authorization rules lay out the tasks that a manager is permitted to perform as well as those that they are not. They restrict the information that is given to managers as well as the actions that they are permitted to conduct in relation to the objects that are managed (see Figure 1). The obligation policies of an organization serve as a guide for decision-making since they explain what a manager must or must not do. The manager is responsible for interpreting these rules in order to achieve the broader goals of the business.

Distributed system management refers to the operations that must be carried out in order to guarantee that large, decentralized networks will function in a manner that is congruent with the objectives of its users. Typically, these objectives are outlined in policies, which are subsequently interpreted by system managers. There are benefits to be gained by either automating mundane administrative tasks or providing human administrators with automated assistance. It is desirable, in order to accomplish this goal, to provide a model of policies as objects, which the framework itself may then represent. Because human administrators might be able to find a peaceful solution to these problems, an automated framework needs to

investigate the many kinds of disagreements that could occur before it can identify them and find a suitable solution to them. There are two distinct classes that policies can fall into. The first set of rules, which are together referred to as authorization policies and obligation policies, detail the actions that a manager is authorized to take as well as the duties for which they are responsible. Policies are presented in the form of objects, and they describe the relationship that exists between subjects, who are referred to as managers, and targets, who are referred to as managed objects. Make use of domains to organize the different items into which a policy might be applied. Second, obligation policies offer direction for decision-making by laying out what a management must and must not do. This helps to ensure that appropriate actions are taken. The manager is responsible for interpreting promises in order to ensure that the organization's primary goals are met.

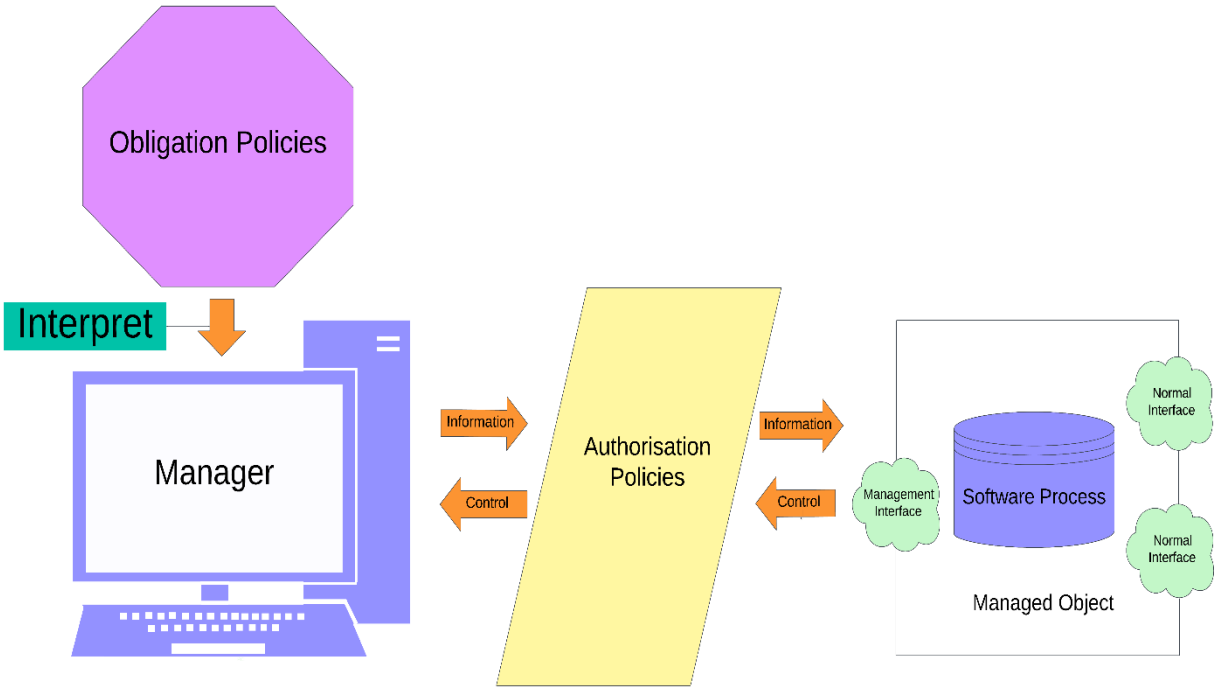

Figure 1: Policies Influence Behaviour

When making judgments, human managers are skilled at interpreting both formal and informal policy specifications and, when necessary, resolving conflicts. However, there is a tendency to automate many management functions into distributed components due to the scale and complexity of big distributed systems. If the policies are hardcoded into these components, they become rigid, and the only way to change how they behave is to recode them. Since management components cannot be changed dynamically, it is necessary to express, represent,

and manipulate policy information independently of them in order to reuse these components for multiple policies. Multiple human managers may establish distinct policies at the same time in relation to the management of a sizable distributed system. Since it is hard to prevent conflicts and inconsistencies due to the complexity of the issue, the policy service must provide analysis in order to identify them and at the very least alert human users to probable conflicts and inconsistencies. The two Esprit-funded projects SysMan and IDSM, which are creating distributed management applications based on the use of domain and policy services, are presented as using common policy principles in this paper.

Managers should be able to make timely management decisions in an integrated management environment. Large heterogeneous distributed systems' complexity and volume of information must be addressed by managers if they are to be effective. System management has evolved into a challenging undertaking due to the abundance of manageable resources, the jumble of administrative strategies, inconsistent tools, and subpar facilities.

2. Review of Literature:

The activities required to ensure that broad distributed networks will operate in compliance with their users' goals are referred to as distributed system management. These goals are usually stated in the form of policies, which are then interpreted by system administrators. There are advantages of offering automatic assistance to human administrators or automating repetitive management functions. It is desirable to provide a model of policies as artefacts that can be represented by the framework itself in order to accomplish this. This is a summary of the model. There is no doubt that policy conflicts will arise. Human administrators may be able to handle these disputes informally, so in order for an automated framework to recognise and resolve them properly, it must first analyse the forms of dispute that may arise. We examine the different forms of policy overlap that may exist to explain how this study relates to the different types of policy dispute. This study is placed in the light of other work on policy, authority and similar fields, including deontic reasoning, and several alternative approaches to dispute prevention and resolution are proposed.

Abstract—Modern distributed systems contain a large number of objects and must be capable of evolving, without shutting down the complete system, to cater for changing requirements. There is a need for distributed, automated management agents whose behaviour also has to dynamically change to reflect the evolution of the system being managed. Policies

are a means of specifying and influencing management behaviour within a distributed system, without coding the behaviour into the manager agents. Our approach is aimed at specifying implementable policies, although policies may be initially specified at the organizational level (c.f. goals) and then refined to implementable actions. We are concern with wo types of policies, Authorization policies specify what activities a manager is permitted or forbidden to do to a set of target objects and are similar to security access-control policies. Obligation policies specify what activities a manager must or must not do to a set of target objects and essentially define the duties of a manager. Conflicts can arise in the set of policies. For example, an obligation policy may define an activity which is forbidden by a negative authorization policy; there may be two authorization policies which permit and forbid an activity or two policies permitting the same manager to sigh checks and approve payments may conflict with an external principle of separation of duties. Conflicts may also arise during the refinement process between the high-level goals and the implementable policies. The system may have to cater for conflicts. such as exceptions to normal authorization policies. This paper reviews policy conflicts, focusing on the problems of conflict detection and resolution. We discuss the various precedence relationships that can be established between policies in order to allow inconsistent policies to coexist within the system and present a conflict analysis tool which forms part of a role-based management framework. Software development and medical environments are used as example scenarios in the paper.

Separating management policy from the automated managers which interpret the policy facilitates the dynamic change of behaviour of a distributed management system. This permits it to adapt to evolutionary changes in the system being managed and to new application requirements. Changing the behaviour of automated managers can be achieved by changing the policy without have to reimplement them – this permits the reuse of the managers in different environments. It is also useful to have a clear specification of the policy applying to human managers in an enterprise. This paper describes the work on policy which has come out of two related ESPRIT funded projects, SysMan and IDSM. Two classes of policy are elaborated – authorisation policies define what a manager is permitted to do and obligation policy define what a manager must do. Policies are specified as objects which define a relationship between subjects (managers) and targets (managed objects). Domains are used to group the objects to which a policy applies. Policy objects also have attributes specifying the

action to be performed and constraints limiting the applicability of the policy. We show how a number of example policies can be modelled using these objects and briefly mention issues relating to policy hierarchy and conflicts between overlapping policies.

The resource management system is the central component of distributed network computing systems. There have been many projects focused on network computing that have designed and implemented resource management systems with a varsity of architectures and services. In this paper, an abstract model and a comprehensive taxonomy for describing resource management architectures is develop ed. The taxonomy. is used to identify approaches followed in the implementation of existing resource management systems for very large-scale network computing systems known as Grids The taxonomy and the survey results are used to identify architectural approaches and issues that have not been fully explored in the research.

Device failures, performance inefficiencies, improper allocation of resources, security compromises, and accounting are some of the problems associated with the operations of distributed

systems. Effective management requires monitoring, interpreting and controlling the behaviour of the distributed system resources, both hardware and software. Current management systems pursue a platform-centred paradigm, where agents monitor the system and collect data, which can be accessed by applications via management protocols. Some of the fundamental limitations of this paradigm include limited scalability, micromanagement, and semantic heterogeneity. We propose an alternative model, Management by Delegation, and contrast its properties via an application example, evaluating the health of a Distributed System.

3. Overlapping Policies:

Within the framework of our model of management intervention policies, groupings of objects are used to depict not just themes but also goal items. The overlap connection is said to exist when the intersection of two sets of points yields a value that is not zero, as seen in figure2

International Journal of Academic Research & Technology (IJART)
Volume: 01 Issue: 01

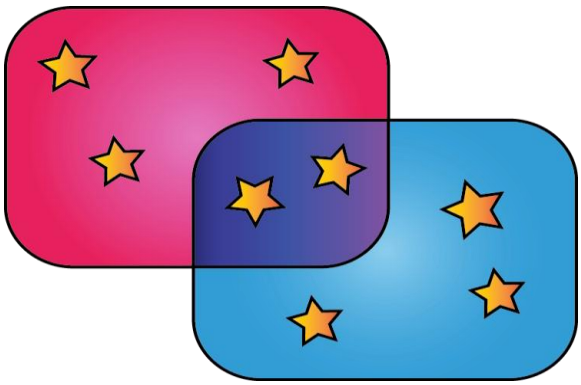

Figure 2: Overlapping sets

We are of the opinion that if there isn't some kind of connection between the topics that are covered by two laws, then there won't be any kind of conflict between them; therefore, overlap is essential to our discussion of policy disputes. When it comes to the conflict analysis that comes later, the first stage of classification that we utilize is the sort of overlap. There are a variety of possible combinations of overlap between policies as a result of the numerous ways in which subjects in the topic and target object sets of the policies might overlap with one another.

- **Double overlap:** There is duplication in terms of both the policies' subjects and their target objects.
- **Subjects - Targets overlap:** The topics of one policy and the aim items of another policy are not the same. For some kinds of conflict-free partnerships between police departments, overlap of any kind, in whatever form it may take, is a necessary prerequisite. Listed below are some examples of policy purpose entity attributes that have similarities but do not contradict one another.
- **Authority hierarchies:** In many different kinds of organizations, the chain of command is clearly laid out. Whenever a higher-level management passed down authority to lower-level managers, the target items were typically broken up into subgroups and assigned to managers in separate departments. There is an obvious duplication of purpose between the objective object set in the policy that provided authority to the higher-level manager and the target object sets that were given to the hierarchical managers to oversee. Even in the extremely uncommon case that a subordinate manager has a breakdown, a higher-level boss is not able to assert power over the priorities that have been given thanks to an imperatival method.

- **Imperatival policy hierarchies:** In many different kinds of organizations, the chain of command is clearly laid out. Whenever a higher-level management passed down authority to lower-level managers, the target items were typically broken up into subgroups and assigned to managers in separate departments. There is an obvious duplication of purpose between the objective object set in the policy that provided authority to the higher-level manager and the target object sets that were given to the hierarchical managers to oversee. Even in the extremely uncommon case that a subordinate manager has a breakdown, a higher-level boss is not able to assert power over the priorities that have been given thanks to an imperatival method.
- **Responsibility:** There is a distinction to be made between being responsible for and having an obligation to complete a task in relation to achieving a goal. We are of the opinion that it would be good to divide the principle of obligation into two separate imperative policies that each refer to the same group of goal issues. The manager who is responsible for achieving the goal, as well as the manager who is responsible for supervising the previous management, will be the focus of both of the aforementioned ideas. Ongoing research pertaining to this topic is being carried out right now.

4. Performance evaluation:

- **Distributed System Management**

The process of monitoring and controlling the functions that are necessary for a framework is referred to as distributed system management. In order to deal with uncertainty and make this practicable for heterogeneous systems, a number of management specifications based on the Open Systems Interconnection (OSI) standard have been established. They partition the overall administration into distinct functional areas with individual responsibility. Management of configuration, management of performance, and management of faults are the three key functional areas that OSI has highlighted as being important. Controlling the installation of both hardware and software components within a distributed system or application is the responsibility of configuration management. The purpose of performance

management is to optimize performance in order to either reduce operating costs, boost throughput, or improve reaction times and reliability. Even though it is not a common component of OSI Systems Management, monitoring information regarding the condition, failures, output, and use of resources is essential for performing any of the management duties described above. Managing distributed systems is similar to managing any other type of organization since it requires adhering to general rules as much as possible rather than following guidelines that are relevant to the situation at hand. the breaking down of management domains into machine objects that can have policies applied to them.

- **Domains**

Domains give the structure for dividing management responsibility. This is accomplished by grouping items for the purpose of describing a management policy or for any other reason a manager may wish. The term "management domain" refers to a collection of controlled elements that have been brought together for the express purpose of facilitating management. A managed object known as a domain is one that maintains a list of links to the managed objects that are part of its membership. A domain is said to be an object's parent if the object has a reference to that domain, and the object is said to be a direct member of the domain to which the reference points. Because of the fact that it is a managed object in its own right, a domain can be part of another domain and is then regarded as being a subdomain of the domain to which it belongs. By segmenting a large collection of objects into fluid subdomains, a number of managers can be delegated the job of applying distinct policies to distinct subgroups of the objects. Members of a subdomain are considered to be indirect members of the parent domain. The concept of a domain is analogous to that of the directory that may be found in hierarchical file systems.

- **Management Policies**

Rules are, according to one definition in the dictionary, the tactics that an organization employs in order to achieve its goals. Every organized organization includes rules. They are the impetus behind the management's decisions. They serve two purposes: first, they determine the priorities of the organization, and second, they determine how much money should be spent to attain those priorities. The policies are implemented as a method of management in a manner that is rather bureaucratic. A policy at a higher level will lead a boss, and the boss will have the ability to accomplish their aims by developing policies at a lower level that will have an effect

on other managers further up in the hierarchy. The vast majority of organizations have policy statements that can provide its representatives with direction in a variety of different scenarios. The policies of an organization can either place constraints on the manner in which the goals should be carried out or offer constructive criticism on the organization's goals and the manner in which they should be accomplished. These kinds of policy statements allot (or grant permission to access) the resources that are necessary to carry out the aims. Because they require the expenditure of resources, we refer to them as budgets.

- **Policy Classification**

When using an object-oriented methodology, the way in which an object interacts with other things in its environment is determined by the object's external behaviour. We further clarify the idea of policy by defining it as the information that has an impact on the interactions that take place between a subject and a target. As a result, the policy outlines the nature of the connection that exists between the subject and the target. Because an object can be the subject or target of several policies, it is possible for multiple policies to apply to that object.

- ***Authorization Policies***

A subject is only allowed to engage in certain activities if the subject's authorization policy specifies those activities in terms of the operations that the subject is permitted to carry out on the target object. An authorization policy can be either positive (permitting) or negative (prohibiting), which means that not permitted equals forbidden in general. Authorization policies are regarded to be target-based when there is a reference monitor connected with the target. This reference monitor is responsible for enforcing the policy and determining whether or not a particular activity is allowed or banned.

- ***Obligations Policies***

The obligations policy specifies the tasks that a subject must complete (or must not complete). The fundamental presumption is that all subjects are well-behaved and will make a sincere effort to comply with policies in which they have no say or agency in the matter. This might be the case with automated subjects, but it won't be the case with human subjects in the vast majority of cases. Policies regarding obligations are said to be subject-based when it is the subject's responsibility to interpret the policy and carry out the activity that has been stated.

- **Policy Attributes**

International Journal of Academic Research & Technology (IJART)
Volume: 01 Issue: 01

Modality, policy themes, policy objective items, policy priorities, and policy boundaries are all features of policies, regardless of whether they are concerned with imperatives or power. Modality can be defined as the degree to which something can be changed. Figure 3 presents an example of how we illustrate policies by employing a common graphical convention (without imposing any limits). For the sake of graphical simplicity, the themes and aim objects are presented using the standard Venn diagram approach. On the other hand, the collection of priorities is presented as a list that is tied to the policy modality. The circles used to symbolize the subjects, and the triangles used to represent the goals, are shown below.

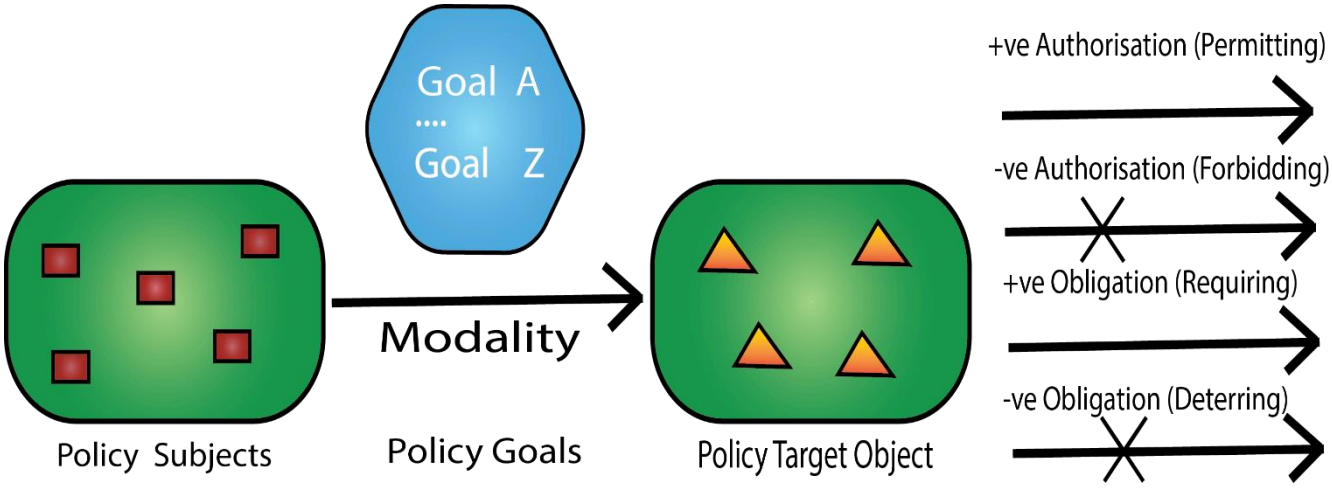

Figure 3: A Management Action Policy

- **Modality**

The four modalities of a regulation (deterring) are positive authority (permitting), negative authority (forbidding), positive imperatival (requiring or obliging), and negative imperatival (forbidding). Positive authority allows something, whereas negative authority prohibits it. These are sufficient for our needs, although we do not rule out the possibility that further helpful policy mechanisms will be suggested in the future.

- **Representing management action policies as objects:**

It is useful to view management action policies as objects on which operations can be performed. For simplicity we assume the following minimal set of operations:

- Create a policy
- Destroy a policy

- Query a policy.

It is possible that you will need authority in order to carry out operations on policy properties. If the computer system is being used solely as a tool to assist with the documentation process, it is possible that no constraints are required. On the other hand, if the rules are actually used to impact the behaviour of the device, such as in the case of access control policies, then actions must be restricted. When policies are represented as explicit objects that managers can access, it is easier to decide which policies are implemented and how to alter them. This is because selecting which policies are implemented is based on which policies are represented as explicit objects. If it becomes essential, read-only versions of the policies can be created to prevent them from being changed. On the other hand, the policy specifications of many programs are encoded into the management or execution components of the system. Even if embedding a protocol within an implementation is the most practical approach to enforce such rules, a (high level) policy object should also be included to explain the policy. This is necessary to ensure that the policy is not updated for a new device update without the user being aware of the change.

5. Research and Methodology:

- **Policy Conflicts**

Policy conflicts can be defined using a few different well-known words. When one person oversees the operations of two separate companies, there is a potential for a conflict of interest because it is more difficult to act ethically when juggling the responsibilities of both companies. In order to avoid violations of the control principle of division of duties, which stipulates that there must be at least two distinct parties participating in the carrying out of essential transactions, conflicts of responsibilities must be avoided. When the available resources are not sufficient to meet the demands that are placed on them, this can lead to a conflict of interests developing. These other, more primitive forms of confrontation are often avoided by human administrators whenever possible, such as in situations in which an activity is both permitted and prohibited, or when someone is obliged to do an action that is prohibited. Human resources administrators use a combination of methodical and intuitive concepts as well as informal interaction in order to identify, avoid, and resolve conflicts. They don't even come close to doing a good job. It is anticipated that automated procedures would adhere to a much more formal strategy due to the fact that the equipment will malfunction if issues are not

managed effectively. This study investigates whether or whether it would be suitable to apply our policy model to the examination of disagreements with the intention of avoiding them, acknowledging them, and resolving them.

- **Policy Constraints**

It is possible for a constraint to be inserted in the policy specification, which would allow the applicability of the policy to be restricted. It is called a predicate and it refers to universal qualities like action parameters or time parameters.

- **Policy Goals**

The policy targets' qualities can be stated as a high-level agenda that specifies what the planner can do in broad terms but does not specify how to achieve the goals. This agenda governs what the planner is able to do. Alternately, the goals could be broken down into a series of more precise tasks that lay out in greater detail how to achieve the intended result. Actions are specified using an alphabet of operations that can be carried out on device properties, so enabling them to be automatically interpreted. These operations can be performed on the properties of the device. One overarching goal can be partitioned into several subgoals and subgoals into many individual action steps. The process of refining a target down to a set of behaviours is analogous to the process of refining a set of parameters into the thorough specification of a computer program.

- **Access Rules**

An access rule is a straightforward illustration of a management authorization policy. This type of policy establishes a relationship between managers (in a subject scope domain) and managed objects (in a target scope domain) in terms of the management activities that are allowed to be performed on objects of a particular type. In addition to defining limitations on these actions, the access rule may also employ scope expressions to pick subsets of the objects that fall under either the subject domain or the target domain. Figure 4 shows that the operations OpA and OpB can be performed on items of type T1, while the operations OpX and OpZ can

be performed on objects of type T2. These processes are only permitted to be carried out between the hours of 8:00 and 16:00.

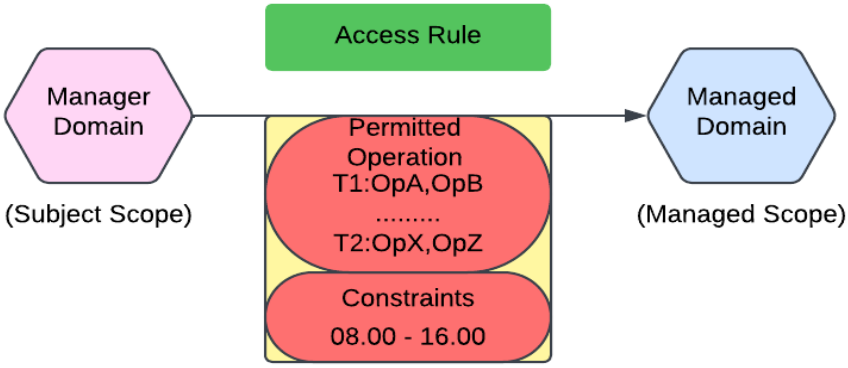

Figure 4: Access Result

6. Analysis and Interpretation

➢ Data Analysis

The performance figures that were acquired from each iteration of the algorithm provided us with information from which we could extrapolate certain conclusions while we were conducting an analysis of a summary of the metrics that were derived from the full dataset. In this section, we reviewed the overall conclusions drawn from the data, as well as the performance of the algorithms, their correctness, and how easy it was for the administrator to utilize their system. General Remarks Regarding the Seed Information for the Runs It was able to compare the two algorithms' performance and accuracy using the same inputs because both algorithms used the same data when they ran their simulations. as an illustration, one of the runs was performed utilizing a user's own custom ACP. In addition to the access that was granted to the user directly in situations where there was no role, it displays the majority of the ACPs for users who were given roles, which in turn provided them access to objects; a sample of conflicting ACPs was taken and is displayed in Table 1 below.

Object Id	Object Name	Access Mode	Is Auto Resolved	Position Rank	Position Name	Role Name	Hierarchy Level
97	Object - 97	Deny	2	1	Executive	Role - 34	2
97	Object - 97	Full	2	1	Executive	Role - 73	3

Table 1: Example of APCs in conflict

The data that is displayed in Table 1 was extracted from an ACP list that belonged to a single user instance that subscribed to multiple roles. It was presumed that the real person to whom this user instance was assigned would only require a single user account to log in whenever they accessed this user instance. The ACP shown in Table 1 revealed that the user is subscribed to two roles; nevertheless, the access to the object was inconsistent between the two roles, which resulted in an ACP conflict.

Metrics were utilized in their respective runs by both of the algorithms that were utilized in the investigation. A list of these is provided down below in Table 2. Because the performance of the algorithms was supposed to be a part of the process of authorizing users to use a system, any delay caused by the execution of the algorithms was added to the process of authorizing users to use the system.

Metric	Value
Number of Runs	100
Total number of ACP processed	24,644
Average ACP per run	246
Minimum ACP count in the runs	65
Maximum ACP count for the runs	984
Median of the ACP count for the runs	182

Table 2: Common Metric Summary Calculated from The Runs

7. Result and Discussion

Because it is impossible to rely on subjects to carry out their requirements, authorization policies are often implemented through the security techniques of the operating system in order to protect target objects. Obligation policies may be implemented either by the management system itself or by the managers who must be trusted. When compared to obligation regulations, the flexibility of authorisation policies is significantly lower. For instance, obligation policies might be activated by an event that results in a certain action being carried out, but they would continue to be inactive until the occurrence of the event once more. According to the findings of our investigation, event-based authorization regulations are not required. Even while a negative obligation and a negative authorization can appear to be the

same thing at first glance, the subject, and not a target-based reference monitor, is the one who is responsible for putting a halt to the activity. This assumes that the individual in question is trustworthy and deserving of respect. The negative duty is engaged in order to temporarily prevent the activity, even though the person may really be authorized to carry it out. This is done even when the behaviour could be considered illegal. For instance, a standby manager may ordinarily be granted permission to carry out control operations, but a negative duty would prevent the backup manager from doing so. Converting a negative obligation into a negative authorization may be conceivable, but doing so may be difficult due to the limits and administrative costs associated with applying authorization requirements. Although this may be possible, it may be difficult to do so. If it is determined that the subject cannot be trusted to execute a negative duty, then the policy must be changed to one that authorizes actions.

- On boiler. temp < 52 controller must switch on boiler heater
- On temperature > 98 controller must switch off boiler heater Negative state-based policies can sometimes be transformed into positive ones by modifying the predicate. For example
- The operator is permitted to close valve on reactor when reactor. Temp ≤ 100 is equivalent to the negative.
- This assumes there is an implicit negative authorisation policy forbidding any access unless a positive authorisation policy permits it. Combining positive and negative policies can result in conflicts

8. Conclusions

It has been discovered that the numerous ways in which policies might overlap with one another have a direct connection to the natural classifications into which they fall. If any future or existing conflict needed human engagement, the majority of the benefits of automation would be lost, and the long-term success of automated distributed system management would depend on the automation of policy dispute identification and resolution. However, before this concept can become a reality, modifications need to be made on at least three different fronts. To get started, it is essential to have a good understanding of the inner workings of the software used for distributed system administration. In this essay, our examples are constructed on what is obvious and customary in their respective fields. By gaining an

understanding of the actual tensions that exist, we will be able to pinpoint the regions that require the greatest effort. Second, there is a need for improvements in the overall formalization of policy. Whether through the use of formal logic reasoning or modelling models, it needs to be systematically approached in order to be dealt with in an appropriate manner. In addition to this, it must be universal, and any policy conflicts that need to be settled must do so in a manner that is amenable to all parties. Third, there is a requirement for the creation and development of operational applications for the theoretical model. Even if initiatives like Domino have begun working on this problem, there is still a significant distance to travel before it becomes profitable.

9. References

1. Bouzida, Y., Logrippo, L., and Mankovski, S. (2011). Concrete- and abstract-based access control. *International Journal of Information Security*, 10(4), 223-238.
2. Grompanopoulos, C., Gouglidis, A., & Mavridou, A., (2021), "Specifying and verifying usage control models and policies in TLA+". *International Journal on Software Tools for Technology Transfer,* 1-16.
3. J.D. Moffett and M.S. Sloman, Policy Conflict Analysis in Distributed Systems Management t, *Ablex Publishing Journal of Organizational Computing*, Vol. 4, No. 1, 1994, pp 1-22.
4. K. Becker and D. Holden, Specifying the Dynamic Behaviour of Management Systems, *Journal of Network and Systems Management*, Vol. 1, pp. 281-298, Sep. 1993, Plenum Press.
5. K. Venkatasalam and M. Pandiyan, Inspecting the Policy Conflicts in Distributed System Management, *International Journal of Production Technology and Management (IJPTM)*, 10(2), 2019, pp. 101–109.
6. Katsikogiannis G., D.Kallergis, Z.Garofalaki, Mitropoulos S., Douligeris C., (2018), A policy-aware Service Oriented Architecture for secure machine-to-machine communications', *Journal of Ad Hoc Networks*, pp. 70-80, November 2018, Elsevier Science Publishers.
7. Moffett, J. D. and Sloman, M. S. (1994). Policy Conflict Analysis in Distributed System Management. *Journal of Organizational Computing*, 4(1), 1-22.

8. Muppavarapu, V., Pereira, A. L., and Chung, S. M. (2010). Role-based access control for a Grid system using OGSA-DAI and Shibboleth. *Journal of Supercomputing*, 54 (2), 154-179.
9. Neela Madheswari A, "The availability of Workloads for Grid Computing Environments", *International Journal of Engineering Research and Technology*, p.no. 211-213, 2015.
10. Lu Y., Fu Q., Xi X., Chen Z., Zou E. and Fu B. (2019), "A policy conflict detection mechanism for multi-controller software-defined networks." *International Journal of Distributed Sensor Networks*. https://bit.ly/2YyIzMP (accessed 5/5/2020).
11. R. Wies, Policies in Network and Systems Management — Formal Definition and Architecture, *Journal of Network and Systems Management*, Vol. 2, No.1 pp.63-83, March 1994, Plenum Press.