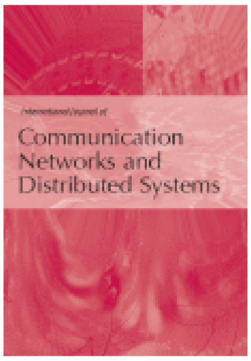


International Journal of Communication Networks and Distributed Systems

ISSN online: 1754-3924 - ISSN print: 1754-3916
https://www.inderscience.com/ijcnds

Developing policy hierarchies for an effective distributed systems and network management: a case study in a videoconference service

Sarandis Mitropoulos

DOI: 10.1504/IJCNDS.2024.10054947

Article History:

Received:	28 October 2022
Last revised:	16 January 2023
Accepted:	16 January 2023
Published online:	30 November 2023

Developing policy hierarchies for an effective distributed systems and network management: a case study in a videoconference service

Sarandis Mitropoulos

Ionian University,
Filosofon and Tzeveleki St., 31100, Lefkada, Greece
Email: smitropoulos@ionio.gr

Abstract: Large-scale distribution systems and network management depend on complex relationships that are strongly related to the managerial roles, which depend on the management policies set. Management policies express peer-to-peer or hierarchical management relationships which constitute the policy hierarchies. Policy hierarchies must satisfy at the highest management level, the business goals, while the complex relationships between policies must be analysed for avoiding conflicts, developing in parallel synergy, strategic convergence, and optimisation of policy usage. This paper proposes an integrated approach for constructing policy hierarchies. A generic policy refinement framework is presented as a crucial factor for a successful policy hierarchy construction. The policy hierarchy analysis functionality is defined, providing a respective software tool in Prolog. The high applicability of the proposed approach is depicted in a case study concerning the management of a videoconference service.

Keywords: integrated distributed system and network management; management policy hierarchies; policy refinement and analysis; videoconference service provision.

Reference to this paper should be made as follows: Mitropoulos, S. (2024) ‘Developing policy hierarchies for an effective distributed systems and network management: a case study in a videoconference service’, *Int. J. Communication Networks and Distributed Systems*, Vol. 30, No. 1, pp.85–113.

Biographical notes: Sarandis Mitropoulos is a Professor of ‘Management Information Systems and Security and Service Management’ in the Regional Development Department of Ionian University. He has also taught and conducted senior research in the Department of Informatics of the University of Piraeus since 2004. He has recently become a Collaborating Scientific Staff of the Hellenic Open University. From 2002 to 2018, he was a special scientist in Informatics (System Analyst) in a bank supervised by the Ministry of Finance of Greece. He received his Diploma and PhD as an Electrical and Computer Engineer from the National Technical University of Athens. He holds an executive MBA degree from the Athens University of Economics and Business. He has also extended experience in the private sector working in ICT research and/or development projects.

1 Introduction

Distributed system management is a quite complex task having as its primary goals high system availability, reliability, and performance. The management of large-scale distributed systems and networks must cope with the heterogeneity and the complexity of such systems. These requirements drive the need for efficient management mechanisms. Management policies have been proved to be a very efficient approach towards this goal. On the other hand, several important issues arise in this approach, such as policy synergy, policy conflict detection and resolution, policy set minimisation and consistency, policy enforcement, etc. Since policies derive from high-level business goals, a coherent refinement of the business goals to lower-level operational policies up to automated system management rules is very important (Mitropoulos, 2022; Dohndorf et al., 2011).

Policies refer to domains that are groups of organisational entities or information technology and communication (ITC) components which define the sphere of influence of managers. The managers are responsible for the policy enforcement onto domains. Management domains are closely related to the management structures providing a flexible means for defining various kinds of management structures, such as hierarchical, cross-functional or networked. Management structures are closely related to the definition of management roles. A management role consists of a set of authorisation and obligation policies. Managers (or manager positions) that fulfil a role must enforce the role's obligations constrained by the role's authorisations. Business goals are reflected in the ICT strategy and in the organisational infrastructure that subsequently drive the definition of the management structures, the manager positions, and the management roles. Analysis, modelling, and discussion on the subject can be found in ISO/IEC JTC1/SC21 (1995) and Rubio-Loyola et al. (2006).

The described management framework implies the development of a set of policies, consisting of peer-to-peer or hierarchical management relationships that create policy hierarchies. Suitably created policy hierarchies prove to be very useful in managing organisations, applications, systems, and networks (Mitropoulos and Douligeris, 2006). Nevertheless, policy hierarchies in large scale distributed systems and networks are quite complex and, thus, need effective and consistent construction, analysis, validation, and optimisation. Towards this direction, several important issues arise, such as policy refinement and validation, policy consistency checking, policy synergies and coordination, policy propagation and delegation of responsibility down through the management hierarchy, policy conflict detection and resolution, policy set optimisation, adaptive policy definition and configuration enforcing policies on policies (meta-policies), etc. (Moffett and Sloman, 1993; Mitropoulos, 2000). Although, there have been some significant research efforts in the policy hierarchy management and analysis, there still exist several open issues and unresolved problems. This paper contributes towards the direction of resolving such problems, specifically, by:

a proposing an integrated layer-based architectural outline for a policy-based management, which is needed for an improved understanding of the role of the various management components, as well as their relationship with the management policy hierarchies.

b proposing a generic framework that supports policy hierarchy construction, as there is a substantial lack of the literature regarding policy hierarchy construction. This framework is role-based and domain-based and provides the means for rendering the various policy relationships.

c proposing a high-level framework for policy refinement, as this is a main process for the initial policy hierarchy setup and population, and, currently, there is a substantial lack of research results on automated task-specific policy refinement. This framework provides an integrated view of organisational and system management, while it depicts the nature of refinement at the various management layers.

d describing the main tasks of a policy hierarchy analysis which is required during the policy hierarchy life cycle from various perspectives. The automation of the policy hierarchy analysis tasks is very important and for this reason this paper presents a partial implementation of a policy hierarchy analysis tool (PHAT) in Prolog, and, finally, by providing a thorough case study in the integrated policy-based management of a videoconference service.

The paper is organised as follows. Section 2 presents an integrated layer-based architectural outline of the policy-based management. Section 3 provides a policy hierarchy definition and construction framework. Section 4 proposes a policy refinement framework (PRF) with respect to the various management levels. Section 5 describes the PHAT main modules along with their implementation in Prolog. Section 6 presents a case study first of the presentation of a videoconference system along with its integrated management, then a service subscription management and the respective policy hierarchy, and finally, an over metropolitan area network (MAN) management providing policy hierarchy definition, refinement, and analysis examples. Several open issues as well as possible future work are discussed in Section 7.

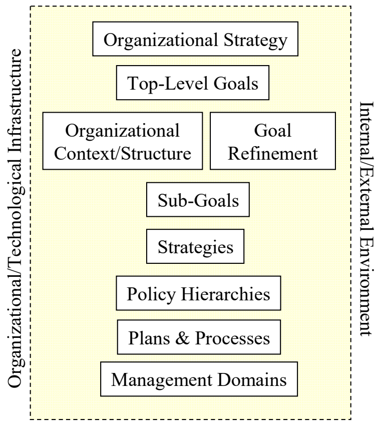
2 An architectural outline of policy-based management

Policy-based management is a method for managing organisational environments, applications, systems, and networks. Policy-based management is related to goal-driven management which is, in turn, related to strategic management (Miller and Dess, 1996; Mitropoulos, 2000). Strategic management is achieved through the enforcement of a set of policies. Policies implement strategies through the development of appropriate plans, processes, and actions. Monitoring mechanisms must provide information related to the policy execution results on the organisation and/or to the ICT infrastructure.

Within an organisation, there are two management viewpoints: the organisational management and the technological management. The goals of organisational management must follow these of technological management. Organisational management must refine the top-level business goals to lower-level goals according to specific criteria imposed by the business context and the internal/external organisational relationships. The high-level goals are refined to sub-goals according to specific goal refinement patterns and rules (Rubio-Loyola et al., 2006). As mentioned in Bandara et al. (2004), "policy refinement problem is composed of refinement of abstract entities into concrete objects/devices and refinement of high-level goals into operations, supported by the concrete objects/devices, that when performed will achieve the high-level goal".

The outline of the above managerial framework is provided in Figure 1 which shows that the initial setup of the organisational strategy is interpreted in specific top-level goals, the interpretation of which into sub-goals is task-specific, namely, depends on the organisational context and structure. These produced sub-goals are satisfied via several strategies that can be achieved by the enforcement of a set of policies which constitute the policy hierarchies. Policy hierarchies concern the management of the operational domains that can be technological or organisational. The management upon them is implemented through plans, processes, and actions.

Figure 1 Architectural outline of the goal-driven/policy-based management



As it is understood from the above analysis, the role of policy-based management is the structuring of the management task using the domain and policy concepts. This approach not only offers system modularity but also provides managers with the capability to enforce different policies on different domains. Furthermore, complex domain structures can be created via various relationships such as containment, overlapping, peer-to-peer, etc. Setting up management domains is equivalent to the specification of the sphere of influence of managers or of the borders of policy applicability (Mitropoulos, 2000; Sloman et al., 1994; Mitropoulos and Veldkamp, 1994). The domains are implemented as system objects, and they are handled by a respective domain service.

Policies are also implemented as system objects and handled by a respective policy service. Policy object attributes concern the policy modality (positive/negative authorisation, positive/negative obligation), the policy subject (managers), the policy target (managed components), the policy actions, the events and the policy execution condition, and the constraints set up for action execution (Mitropoulos, 2000; Mitropoulos and Douligeris, 2006; Damianou et al., 2001; Sloman, 1994). Thus, a Policy Object could be defined as: PolicyObject = {Modality, Subject, Target, Actions, Events/Conditions, Constraints}. A policy object can be formally specified by an object or policy specification language such the ‘Guidelines for Definitions of Managed Objects’ (ISO/GDMO) (ISO/IEC, 1992), the OASIS web services (WS) policy specifications (W3C, 2006), Ponder (Damianou et al., 2001), the security policy language (SPL) (Ribeiro et al., 2001), the policy core information model (PCIM) of IETF/DMTF (Moore et al., 2001), the role-based access control (RBAC) (Bertino et al., 2000; Chen and Sandhu, 1995), the XACML (OASIS, 2001), etc. The subject of a policy is obliged or authorised to do actions onto the policy target. The policy subject is the managers (human or software components) that can also be subject to other policies. From the technical point of view, disengaging a manager from a policy can be realised in a simple way by deactivating the policy or by changing the respective policy subject without the need for recompiling the manager embedded functionality code.

The organisational environment and infrastructure mentioned in Figure 1 concern among others and the underlying distributed system management platform, which must provide policy-based management services such as the policy service, the domain service, the policy enforcement service, the policy monitoring service, the policy analysis, and conflict resolution service, etc. These services use the underlying platform distributed processing services and mechanisms, such as file services, security services, time services, transaction services, distributed object services, etc. for promoting their functionality (Mitropoulos, 2000; Dulay et al., 2001; Agrawal et al., 2005a, 2005b). The distributed management environment (OSF/DME) is an object-oriented infrastructure based on CORBA (Vinoski, 1997). Interoperability can be achieved using RPC/IDL to CMIP/GDMO gateways, with respect to the service interface specifications translation from one language to the other (Forbici and Penna, 1997). Architectural modules for distributed systems interoperability management can be found in Ravuri et al. (2022) and Lytras et al. (2021).

In short, in this section, an integrated framework regarding the hierarchical policy-based distributed systems and network management was presented. This framework provides the platform upon which the definition and the analysis of domains, policies and roles can take place. This is a prerequisite for developing a respective tool which can perform relative tasks. Hereafter, we focus on the analysis of these tasks and a respective tool expanding the related work as fruitfully mentioned above.

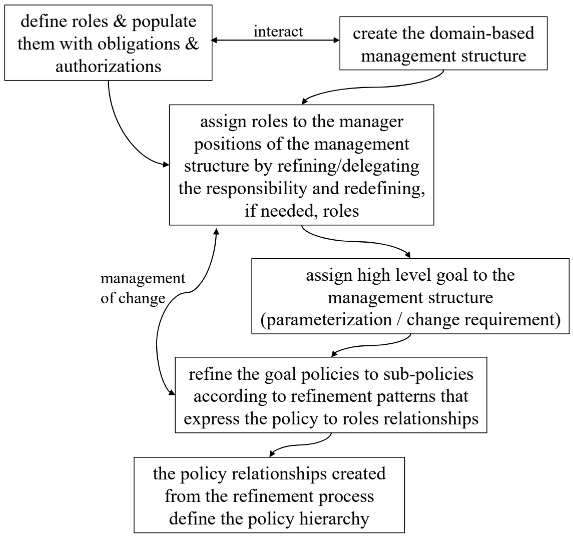
3 Policy hierarchy definition and construction

As explained above, management policy types concern high from top to down level the business goals, organisational viewpoint policies, the system management functional area policies, the system operations policies, and at the lowest level the mechanical rules, like the access control lists. In fact, these policy types follow the hierarchical path of management from the high-level goals up to the low-level mechanical information. The production of lower-level policies takes place through a repetitive refinement process. The policies that stand at a high level of management must be translated and refined to lower-level sub-policies and this process leads to the creation of a policy hierarchy, which usually in large scale distributed systems include complex relationships between managers roles and the respective policies that need analysis from various perspectives.

As already explained, managers (organisational or system) are assigned to roles that consist of a set of policies. Role specification within a management structure and the policy hierarchy construction are closely related. A policy hierarchy provides the means to obtain a coherent understanding of the impact of the defined organisational and system management roles. Roles provide the means for specifying the manager (or manager position) functionality within a management structure, interacting with each other, through several obligations and authorisations that they have towards each other (Lupu et al., 1999).

Let us consider the following example. In a security information management (SIM) system, upon a security incidence occurrence, the incidence response (IR) process starts up involving various corporate parties, each one having its own distinct role and rights in the entire IR process according to the RBAC scenario. Authorisation policies can be assigned to each involved role (or party) to allow this party to access the respective systems or to immediately contact other appropriate parties in an urgent fashion. For instance, a simple user will be logged out after an incidence occurrence, while information disseminators will acquire system access privileges for forwarding messages of emergency, e.g., a security report to the security staff. In fact, the IR scenario of roles' obligations and authorisations creates a sequence of policies within a policy hierarchy, which includes parent/child or peer-to-peer policy relationships.

Figure 2 The role-based policy hierarchy creation waterfall


define roles & populate them with obligations & authorizations
interact
create the domain-based management structure
assign roles to the manager positions of the management structure by refining/delegating the responsibility and redefining, if needed, roles
management of change
assign high level goal to the management structure (parameterization / change requirement)
refine the goal policies to sub-policies according to refinement patterns that express the policy to roles relationships
the policy relationships created from the refinement process define the policy hierarchy

Hereafter, to clarify the phases for the definition of a methodological and coherent policy hierarchy construction, this paper proposes a policy hierarchy creation waterfall as presented in Figure 2. There are two main actions that take place primarily in a supplementary and concurrent fashion: the creation of a domain-based management structure which is, of course, related to the roles that are assigned to the managers of these domains, and the definition of these roles that must be populated with the appropriate obligations and authorisations. The management structure's manager positions must then be assigned with the appropriate roles, or in other words, the manager positions must fulfil specific manager roles. First, a high-level management policy hierarchy is created which must be unfold via the management structure by assigning or refining the responsibility from the high-level managers to the managers of the lower-levels. This process may lead to the redefinition of some manager roles due to inconsistency problems that are discovered during this phase. We assume that a change or configuration requirement sets up a high-level goal policy. This goal policy is refined to sub-policies according to predefined translation rules or refinement patterns. These new policies may cause feedback to the previous phase of refining/delegating the responsibility or redefining the roles. This is the reason why the management of change' arrow is double-sided. The repetitive policy refinement process via the management structure finally creates a policy hierarchy. Every policy produced during the refinement process must be consistent with the already produced policies at the various management levels.

A Policy Hierarchy is formally defined as follows: PolicyHierarchy:= (Policies, Policy_Definitions, Policy_Relationships), where Policies provide the contained in the policy hierarchy policy object names and/or IDs, Policy_Definitions provide the detailed definition of each policy object, while Policy_Relationships describe the hierarchical (or peer-to-peer) relationships between the policy (objects), i.e., {Policy_2, Policy_3} = RefinementOf(Policy_1), which means that policies P2, P3 are produced by P1.

The policy hierarchy creation is a rather hard task, especially in large-scale organisations, distributed systems, and networks because they contain numerous relationships between their manager roles and between their components. In other words, the main problem in policy hierarchies is the identification of the relationships between the policies. These relationships depend on many factors such as the management structures defined in terms of domains, the manager roles and the interactivities between them, the management context refinement patterns and rules, the change requirements, etc. (Rubio-Loyola et al., 2006; Mitropoulos and Douligeris, 2006; Dursun and Üstündağ, 2021).

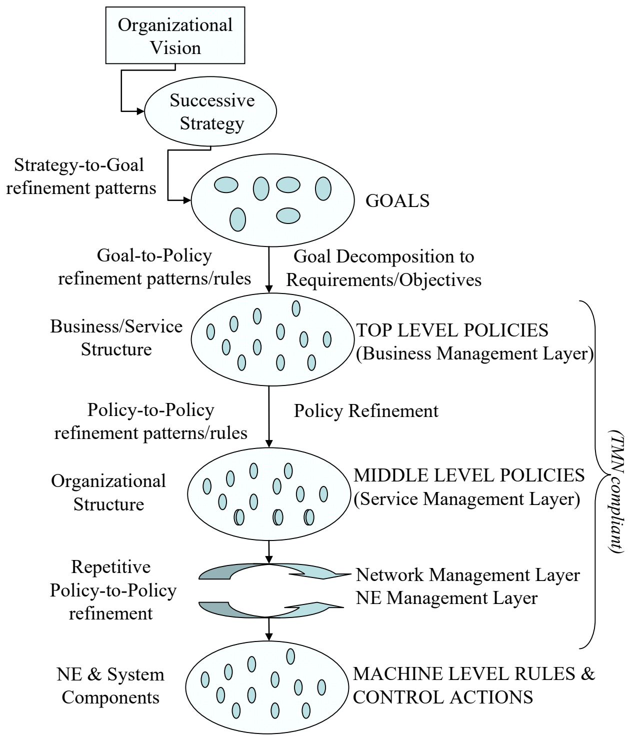
4 A policy refinement network

The policy hierarchy creation is based on refining higher level goals and/or policies to lower-level ones according to specific criteria and refinement rules. In fact, policy refinement is a way of delegating the management task from top to down within a management hierarchy. A likely malfunction of the policy refinement process usually negatively affects the operation of organisations and their systems. At the organisational level, the refinement process needs human expertise and domain knowledge. On the other hand, at the system level the refinement process can be partially automated. Obviously, the result of the policy refinement process and the produced policies must be correct and consistent. According to Bandara et al. (2003), "if there exists a set of policies Prs:P1, P2,..., Pn, such that the enforcement of a combination of these policies results in a system behaving in an identical manner to a system that is enforcing some base policy Pb, it can be said that Prs is a refinement of Pb". In addition, policies specified in Ponder can be translated into an Event Calculus (EC) representation because EC supports the semantics of Damianou et al. (2001) and Lymberopoulos et al., 2003a).

But the question which arises is: "how can refinement be correct and consistent?". In (Bandara et al., 2004), "each high-level goal is refined into sub-goals, forming a goal refinement hierarchy where the dependencies between the goals at the different levels of refinement are based on the form of goal decomposition used (AND/OR)". A top goal can be AND-refined, where the refinement is complete in the conjunction of the sub-goals, or it can be OR-refined into alternative sets of sub-goals. Goal satisfaction can be modelled both quantitatively and qualitatively (Letier and Lamsweerde, 2004; Bleistein et al., 2006). Then, goals are refined to policies until they can be assigned as responsibilities to every single manager. KAOS (Lamsweerde, 2001) is a quite similar formal technique for goal elaboration and refinement. The top goal is disjointed in sub-goals for which multiple strategies are developed, as sets of actions that will achieve the given goals. Abductive reasoning is used to identify strategies that can be used for action clauses of the refined goals or policies (Bandara et al., 2004, 2006; Mitropoulos, 2022).

In this section, an integrated and generic PRF is proposed. The main concerns of this framework are the various phases from top to down of the refinement process as well as the nature of refinement in each phase. As depicted in Figure 3, the organisational vision must be first set up and it must be accordingly translated to a (business) strategy for the organisation. This strategy can be achieved by setting up a set of high level (business) goals. The production of these goals takes place according to specific strategy-to-goal refinement patterns and according to the relevant business and organisational context. These high-level goals must be decomposed into a set of high-level requirements and organisational objectives that can be achieved via a set of high-level policies (Ciavaglia and Peloso, 2019; Mitropoulos, 2021; Ravuri et al., 2022). The production of these high-level policies takes place according to specific goal-to-policy refinement patterns. The high-level policies, which correspond to the TMN's business management layer, will be refined to middle level policies that are mainly concerned with the organisational service provision which among others includes the various distributed computing systems services. The middle level policies will be repetitively refined up to the low-level system policies and machine level operating rules. In other words, the refinement process is enforced from top-to-down (Jiang and Wang, 2023).

Figure 3 The proposed generic PRF based on the various management layers (see online version for colours)


Organizational Vision
Successive Strategy
Strategy-to-Goal refinement patterns
GOALS
Goal-to-Policy refinement patterns/rules
Goal Decomposition to Requirements/Objectives
Business/Service Structure
TOP LEVEL POLICIES (Business Management Layer)
Policy-to-Policy refinement patterns/rules
Policy Refinement
Organizational Structure
MIDDLE LEVEL POLICIES (Service Management Layer)
(TMN compliant)
Repetitive Policy-to-Policy refinement
Network Management Layer
NE Management Layer
NE & System Components
MACHINE LEVEL RULES & CONTROL ACTIONS

Note: The small circles express goal/policy objects while the big circles goal/policy groups.

From the implementation point of view, the policy refinement task is undertaken by top level managers or by an external policy hierarchy refinement supporting tool. In both cases, the policy refinement process consists of two phases. In the first phase, all the incoming policies within a management node (i.e., a manager) must be obtained, pre-processed, analysed and examined regarding their interrelationships. In the second phase, the incoming policies will be refined to lower-level policies that will be assigned to other specific managers for enforcement. A manager (software code or human being) will perform the decision-making process, develop plans, examine alternative solutions, and take several actions under event-driven conditions and constraints.

In this research area, the related work is still at a primitive stage. POWER is an environment which provides policy refinement based on policy templates that support the business policy creation (Casassa et al., 2000). Another framework can be found in (Rubio-Loyola et al., 2006). This framework is based on goal-driven requirement analysis which drives the consultants to define views of management formulating any combination of views and developing policy hierarchies by refining executable policies from the high-level requirements. A similar approach can be found in Bandara et al. (2006) as well.

In conclusion, our proposed PRF first provides an integrated view of the management by putting together both organisational and system management issues, and second it dimensions the various management layers and viewpoints that are useful to both management consultants and administrators for defining and refining policies.

5 Policy hierarchy analysis

As mentioned in the previous section, the produced policies must be correct and consistent with respect to the management goals. For this reason, the policy hierarchy analysis must take place before every policy is enforced. Such an analysis is vital and inevitable in policy-based management systems since otherwise many problems are expected to arise (Lupu and Sloman, 1999; Bandara et al., 2003; Damianou et al., 2002a, 2002b; Bandara et al., 2006; Dursun and Üstündağ, 2021; Ravuri et al., 2022; Clemm et al., 2020; Arzo et al., 2021). Hereafter, the policy hierarchy analysis functionality by specifying the main modules that a PHAT must consist of, are coherently dimensioned. Similar approach can be found in Mitropoulos and Douligeris (2010), Bandara et al. (2004, 2003; Lymberopoulos et al., 2003a; Mitropoulos, 2000; Lupu et al., 1999; Lupu and Sloman, 1999). Thus, PHAT mainly provides analysis on the following:

- *Policy merging* of two or more policies that concern a common goal over a common managed object set. Policy merging is very important especially for cooperative managers that are engaged to achieve a common goal. Formally speaking, if two policies P1 and P2 have common targets T1=T2 and their action set refers to a common interest (AC1, AC2 present similar results) then the two policies can be merged in a new policy P3 by merging accordingly the action sets and by keeping the common target as the target set of P3. Other cases for policy merging that are not mentioned here may arise. Policy merging can be considered as a degenerate case of Policy Synergy which is not presented here for reasons of simplicity.

- *Policy conflict detection* for localising conflicting actions or authorisations over a common target set of managed resources. Formally speaking, if the target sets (T1, T2) of two policies P1, P2 overlap and the action sets of these policies are contradictory in the intersection of the target sets, then the two policies are said to be in conflict (Mitropoulos, 2022; Zhang et al., 2022). Of course, there are also other policy conflict cases, e.g., due to the indirect negative impact of a policy over the activity of another policy within the same policy hierarchy. For example, we assume the introduction of a new security policy the interpretation of which to sub-policies creates an access prohibition policy for the computer room A, something which conflicts with an already produced policy which permits access to computer room A for the same hours. This is an indirect conflict which must be detected, as well. Another policy conflict example is when a manager is obliged to undertake an action without the corresponding authorisation. For example, if the subject set of a Policy P1 is not authorised (by a respective policy) to undertake action over a Target set T1, and the policy P1 obliges the subject to undertake action over T1, then a conflict arises. This conflict is called 'missing obligation authorisation' and will be detected by PHAT executing the CheckMissingObligationAuthorization action which can be formally defined as follows:

  action(CheckMissingObligationAuthorization) => if ((Subject, Modality, ActionSet, Target) && (Modality <- Obligation || Modality<-Permission) && (Modality <- NoPermission) == true) then (notify(MissingObligationAuthorization) => (ListofManagers))

  where =>: action or notification, <-: attribute value assignment.

- *Policy consistency checking* which concerns the localisation of inconsistencies between the results of various activated (dynamic checking) or non-activated (static checking) policies. This case is quite equivalent to policy conflict detection with respect to checking the contradictory influence which a policy has on the system behaviour in relation to the influence of another policy.

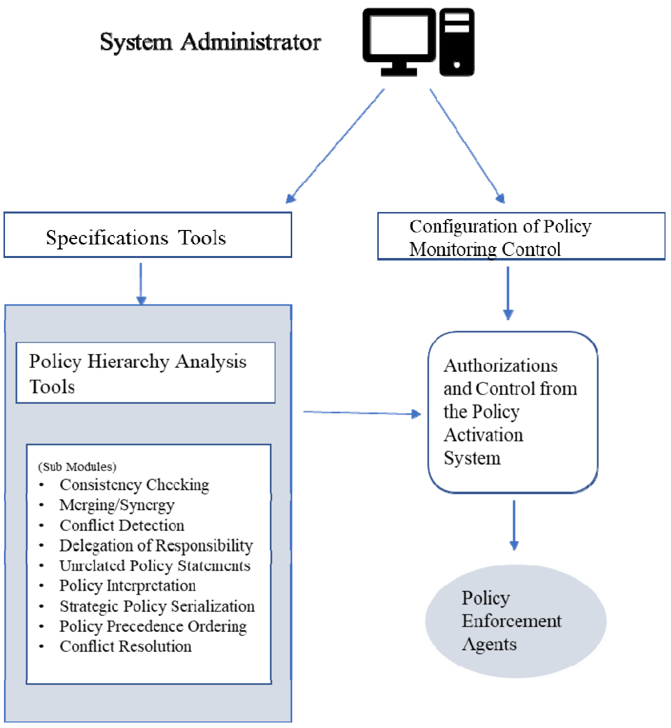
- *Policy conflict resolution (policy precedence ordering)* which concerns reactions that follow the conflict detection. Such reactions include the precedence ordering between competing or in conflict policies, and the creation of alerts and notifications dissemination to the system administrators or to hierarchically upper manager objects that can afterwards go through the activated system policies and investigate how the conflict or the side effects of the conflict in system behaviour can be eliminated. For example, in the 'missing obligation authorisation' example, if the authorisation has the highest priority, then P1 must be disabled, otherwise the respective authorisation must be assigned to the subject of P1 (or equivalently the respective authorisation policy must be created and activated). We note that the Policy Precedence Ordering is not just policy serialisation, which may concern the achievement of a strategic goal, but an ordering of policies with respect to their competence priority. Approaches and mechanisms in resolving policy conflicts in distributed systems and network management can be found in Mitropoulos (2022), Baliosian and Serrat (2004), Charalambides et al. (2005), Kamoda et al. (2005), Agrawal et al., 2005a, 2005b; Lupu and Sloman,1999; Jiang and Wang, 2023).

- *Policy propagation* is the simplest policy analysis case and concerns the propagation of a policy down through the management hierarchy (structure). In policy propagation managers pass-through the hierarchy a policy without any special interpretation. In the partitioned policy propagation, the target set domain is divided into two or more target sub-domains before policy propagation takes place. In both cases, policy propagation is quite similar to the delegation of a manager's responsibility to other managers via policies. Special care must be taken regarding cycles within the domain management hierarchy which via policy propagation may cause unlimited policy action execution. Formally speaking, if there is a policy P1 propagated down through the management hierarchy and its target T1 can be partitioned in two other targets T2 and T3, then create two new policies P2 and P3 with the same action set and targets T2 and T3 respectively (this is the case of partitioned policy propagation to different sub-domains). While if the target set domain T1 includes a sub-domain T11, then create a new policy P11 with the same action set as P1 and target set T11 (this is the case of simple policy propagation to a sub-domain).
- *Goal policy interpretation* concerns a goal-level policy interpretation to sub-policies which apply to the same set of managed objects. The produced policies must be satisfied so that the initial goal to be satisfied. Formally speaking, if a Policy Goal (PG1) has target set T1 and it can be interpreted to more than one policy sets over T1, then substitute PG1 with P1, …, PN that is the equivalent policy set enforced over the initial policy goal target T1.
- *Policy translation* consists of identifying and determining the type of information conveyed by a policy and converting that information into the specified format for the targeted policy level. It should be noted that business-level policies are related to enterprise-level service goals and requests, service-level policies are related to service characteristics as they reflect specific network parameters, and network-level policies are related to required functionality of autonomous mechanisms/components/resources on specific network segments (Galani et al., 2012).
- *Strategic policy serialisation* refers to cases where policies must be enforced in a specific order or strategic plan to achieve a larger goal. Formally speaking, if there is a policy set {P1, …, PN} develop a strategic plan and enforce the policies within the policy set at a specific order and/or at a specific time.
- *Unrelated policy statements* concern the case of two parent-child policies with no identical targets or actions. Formally speaking, if we have the policy P1 then refine P1's actions set (AC1) over its target set (T1) to new actions set (AC2) over a new target set (T2).

From the architectural point of view, the policy hierarchy analysis services are offered at the common management platform services layer which includes among others the policy service, the domain service, the policy monitoring service, the policy enforcement service, etc., and they are provided in our case by the PHAT which obviously must cooperate with the other policy-oriented management services. A comparative architecture of inter-working management service components can be found in Mitropoulos (2000). Figure 4 depicts an architectural diagram of the relationships of

PHAT with other tools and services of a management platform, specifically, with the policy specification tool and the policy activation system. PHAT is useful for both dynamic and static analysis of policy relationships.

Figure 4 The PHAT and its interworking with other tools and services (see online version for colours)


System Administrator
Specifications Tools
Configuration of Policy Monitoring Control
Policy Hierarchy Analysis Tools
(Sub Modules)
• Consistency Checking
• Merging/Synergy
• Conflict Detection
• Delegation of Responsibility
• Unrelated Policy Statements
• Policy Interpretation
• Strategic Policy Serialization
• Policy Precedence Ordering
• Conflict Resolution
Authorizations and Control from the Policy Activation System
Policy Enforcement Agents

Policy-based management represents an automated way of implementing the human-based management task that needs the support of intelligence and expertise. For this purpose, a PHATs must incorporate adequate intelligence and knowledge, but this is not always possible and for this reason the policy refinement and analysis function is difficult to be fully automated. Figure 5 depicts an indicative implementation view of PHAT in Prolog language.

6 Integrated policy based management of a videoconference service

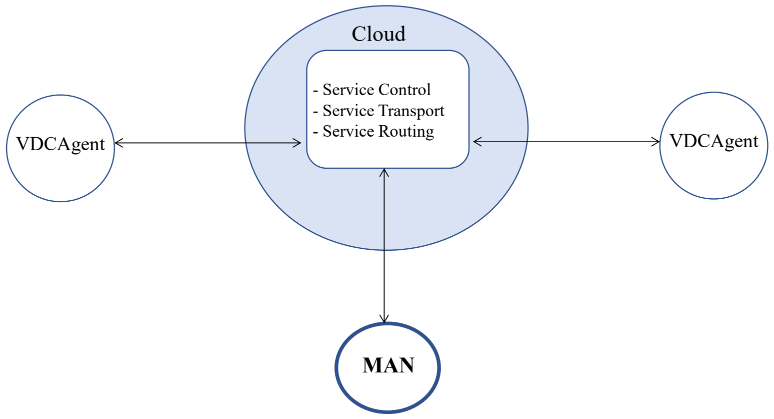
Hereafter, it is presented a case study which concerns the integrated management of a videoconference service over the Cloud, as well as the management of a MAN upon which the service is delivered. Specifically, we first present the videoconference service provision system along with its integrated management architecture. Next, we focus on the management of a service subscription management application providing the respective policy hierarchy. Then, we focus on the MAN network management with respect to initialisations on managers' authorisations and obligations, to quality of service (QoS) and to policy hierarchy definition, refinement, and analysis examples.

Figure 5 The PHAT implementation using PROLOG

```
domains
        Policy = (name, subject, target, actions, constraints)
        Domain = (name, members, parents, children)
        Manager = (name, incoming_policies, outgoing_policies)
        ManagedObject = (name, attributes, actions, notifications)
        PolicyHierarchy = (Policies, Relationships, Descriptions)
        DomainHierarchy = (Domains, Relationships, Descriptions)
databases               /*indicative outline*/
        PolicyRepository(Policy_Objects)
        DomainRepository(Domain_Objects)
        ManagementInformationBase(Managed_Objects)
        PolicyHierarchyRepository(Policy_Objects,Policy_Descriptions,Relationships)
        DomainHierarchyRepository(Domain_Object,Descriptions,Relationships)
        PolicyRefinementRelationships(Policy_Object_Relationships)
predicates
        Refined(.)          ForEvery(.)        member(.)          order(.)  objective(.)
        completed(.)        newpolicy(.)       merge(.)           equivalent(.)      conflict(.)
        ResponsibilityDelegation(.)            Priorities_in(.)   PolicySynergy(.) PolicyConlfict(.)
        PolicyInterpretation(.)                SimplePropagation(.)
        PartitionedPropagation(.)              UnrelatedStatements(.)     Serialization(.)
        ResponsibilityDelegation(.)            PrecedenceOrdering(.)
goal
        PolicyHierarchyAnalysis(PolicyHierarchy).
clauses
PolicyHierarchyAnalysis(PolicyHierarchy):- ForEvery(member(Node,PolicyHierachy),
        PolicySynergy(Node,.), PolicyConflict(Node,.)), PolicyInterpretation(Node,.),
        SimplePropagation(Node.),PartitionedPropagation(Node,.), UnrelatedStatements(Node,.),
        Serialization(Node,.), ResponsibilityDelegation(Node,.), PrecedenceOrdering(Node,.)).
PolicyMerging(Node,policy1,policy2):- objective(policy1,target), objective(policy2,target),
        ForEvery(member(action1,policy1),member(action2,policy2),
        equivalent(action1,action2),merge(action1,action2,newaction)),
        newpolicy(policy,objective(newaction,target)).
PolicyConflict(Node,policy1,policy2):- objective(policy1,target1),objective(policy2,target2),
        overlaps(target1,target2),
        ForEvery(member(action1,policy1), member(action2,policy2), conflict(action1,action2)).
SimplePropagation(Node,policy,target):- objective(policy,target), completed(policy,targetnew).
PartitionedPropagation(Node,policy,target):- objective(policy,target), partition(target,SubList[.]),
        ForEvery(CreateNoOverlapList(SubList[.],NoOverlapList[.]),
        member(subtarget,NoOverlapList), completed(policy,subtarget)).
PolicyInterpretation(Node,policy,target):-
        objective(policy,target), Refined(policy,SubPolicyList[.]),
        ForEvery(member(SubPolicy,SubPolicyList[.]), completed(SubPolicy,target).
UnrelatedStatements(Node,policy,target):-
        objective(policy,target), objective(policynew,targetnew),
        Refined(policy,policynew), completed(policynew,targetnew).
Serialization(Node,policy,target):- objective(policy,target), Refined(policy,SubPolicyList[.]),
order(SubPolicyList[.]),
        ForEvery(member(subpolicy,SubPolicyList[.]), completed(subpolicy,target)).
ResponsibilityDelegation(Node,policy,target):- objective(policy,target),
Part_Refined(policy,subpolicy),
        ResponsibilityDelegation(manager(1),subpolicy,manager(2)).
PrecedenceOrdering(Node,policy,target):- objective(policy,target),
        Refined(policy,SubPolicyList[.]), Priorities_in(SubPolicyList[.]).
member(name, list[name|_]).
member(name, list[_|Tail]):- member(name,Tail).
```

A videoconference system "conducts a conference between two or more participants at different sites by using computer networks to transmit audio and video data".[1] Video as a service (VaaS) in cloud computing can be combined with other cloud services for customised and more flexible management (Radwan et al., 2019). The deployment of the Videoconference service takes place in a platform offered as a service (PaaS) where the management functionality is deployed, as well. Finally, the various servers, endpoint devices and the network constitute the infrastructure which is provided as a service (IaaS). IaaS can scale or shrink the infrastructure on a demand base and according to the service requirements running over it.[2] Cloud service provision, as many other services, must cover control and monitoring services, namely, management services, let us say them 'service control points' (SCP), transportation services, let's say them, 'service transportation points' (STP), and finally switching points for routing the services of the videoconference streams, let us say them 'service switching points' (SSP) (Agrawal, 2021).

Figure 6 Videoconference service provision system (see online version for colours)



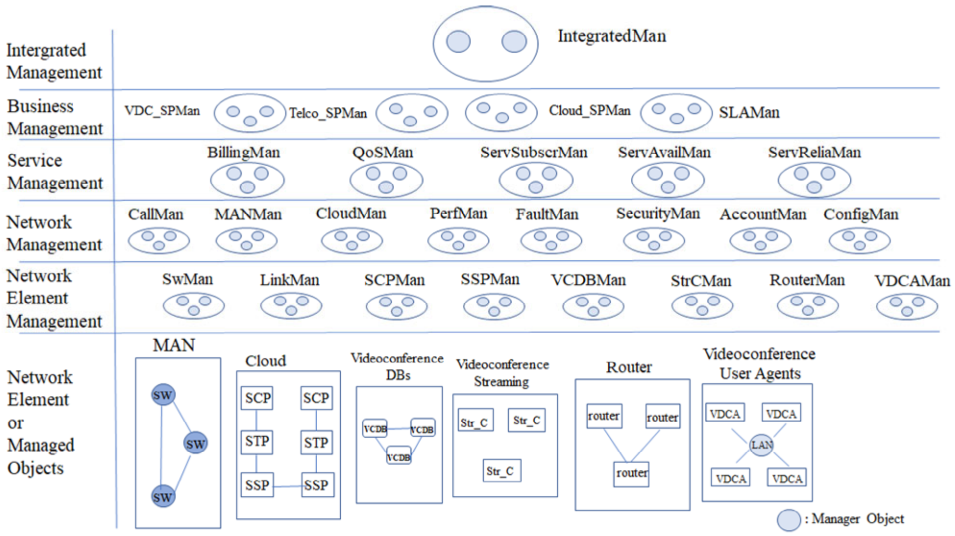
In Figure 6, the start and end points of the lines/arrows may be probably checked again. Also, in the same figure, the border line of the 'MAN' cycle would look better if being more solid.

Management of this service provisioning system obviously concerns various management layers. According to the telecommunications management network (TMN) standard[3], these layers are the business management layer, the service management layer, the network management layer, and the network element management layer. At the bottom layer stand the network elements which are represented as managed objects. Figure 7 provides a complete view for an integrated management of the videoconference service provisioning system including the various manager or managed objects (Mitropoulos, 2000). At the bottom layer, there are the various managed objects.

As depicted in Figure 7, the managed objects (elements) stand at the bottom layer and their management agents one layer above (Ravuri et al., 2022; Zhang et al., 2022; Lytras et al., 2021; Dursun and Üstündağ, 2021). At the network management layer, the managers support the main ISO system management functions, such as performance, fault tolerance, security, accounting, and configuration management along with some other supplementary functions, such as call admission, MAN management which integrates the management of specific MAN components, such as routers, switches and links.

At the service management layer, there are several service-related functions, such as billing, service subscription, QoS management, service availability, service reliability, etc. One layer upper, we have the integration of this functionality with respect to the management supported by the various service providers such as the videoconference service provider, the MAN Operator, and the cloud service provider (Cloud_SP), which may belong in the same management entity, or they may have been outsourced to a third-party provider/operator (Alam et al, 2016). These entities are managed by business related concepts and criteria, and for this reason, they are considered at the business management layer. At the same layer, service level agreement (SLA) is a crucial function which may concern intra-service, outsourcing or customer contracts. At the top layer, managers are responsible for the integration of management of the various involved parties. We note that a manager can consist of various manager object instances something which declares the distributed nature of manager object implementation. These instances are depicted in Figure 7 with small circles in the respective manager domains (big circles). In addition, for simplicity reasons, we assume that each manager position corresponds to a specific role. This management structure was developed through a repetitive refinement process from the top-level roles to the lower-level ones. Of course, there are many and complicated relationships between the manager/managed roles of the same or different management layers, which are not shown for simplicity reasons (Lytras et al., 2021; Dursun and Üstündağ, 2021; Moussa et al., 2022; Jiang and Wang, 2023).

Figure 7 Integrated management of the videoconference service provision system (see online version for colours)



6.1 *Policy hierarchy for subscription service management*

In this section, we focus on a policy-based service management application example that provides a respective policy hierarchy. In Figure 8, an indicative part of the whole management structure, here called videoconference management system (VDC_MS), is formally defined in Ponder (Damianou et al., 2001). The Integrated Manager is assumed for controlling and monitoring the subordinate management systems, which are the Cloud_SPMan, the cloud management system, the VDC_SPMan, the videoconference service provider over the cloud and, the ServSubscrMan, the customer Service Subscription Management System. The respective manager roles are defined together with the relationships between these various roles. Then two instances of the management system are defined with the respective manager positions fulfilling the various predefined roles. As Figure 8 depicts, we assume two management system instances (videoconference management centres, named VDC_SMS) for managing the videoconference service deployed at two different geographical areas (see also Figure 6).

Figure 8 A part of the management structure definition for a videoconference service in PONDER

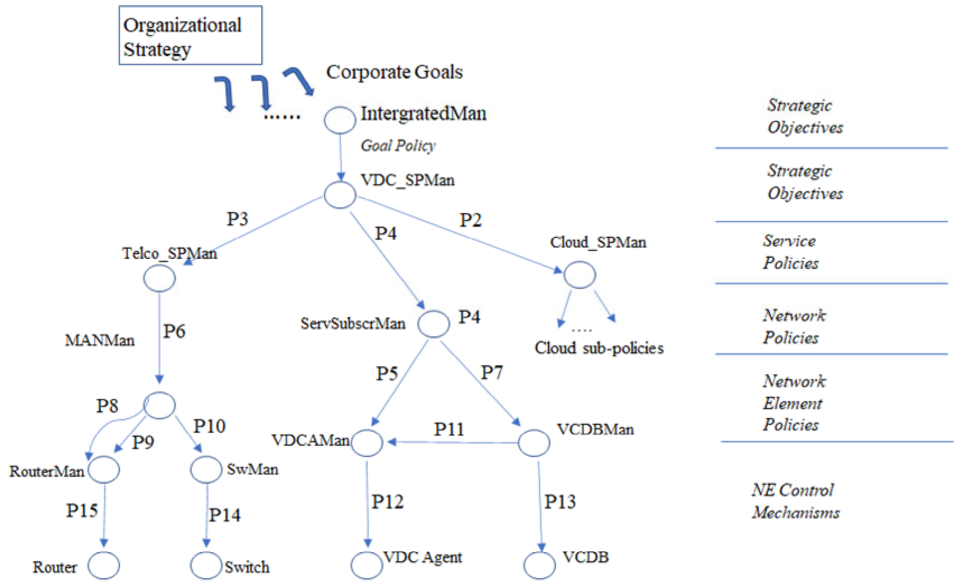
```
type mstruct VDC_SMS(...) {   // VDC_SMS is the Videoconference Service Management System Role
inst role IntegratedManager = IntegratedManagerT(…);
            role Cloud_SPMan = Cloud_SPManT(...);     // Cloud_SPMan is the Cloud Management System Role
      role VDC_SPMan = VDC_SPManT(...); // VDC_SPMan is the Videoconference Service Provider Role
      role ServSubscrMan = ServSubscrManT(...); // ServSubscrMan is the       Service Subscription Manager Role
    // we assume two main management actions: Control & Monitor
inst   rel control = ControlT (IntegratedManager, Cloud_SPMan);
             rel monitor = MonitorT (IntegratedManager, Cloud_SPMan);
             rel control = ControlT (IntegratedManager, VDC_SPMan);
             rel monitor = MonitorT (IntegratedManager, VDC_SPMan);
             rel control = ControlT (IntegratedManager, ServSubscrMan);
          rel monitor = MonitorT (IntegratedManager, ServSubscrMan);
}
inst  mstruct VDC_SMS_A = VDC_SMST(…);
              mstruct VDC_SMS_B = VDC_SMST(…);
```

Having defined the management structure and the respective manager roles, we recursively enforce the policy refinement process from top-to-down according to the PRF framework presented in Section 4. We assume that a successive organisational strategy for customer service and support of high quality among others requires that “the service subscription must effectively take place without problems and delays in order the videoconference service to be available to the users as soon as possible according to the SLA criteria”. The latter can be considered as a corporate goal which sets up requirements and objectives to the overall management system operation. This corporate goal will be satisfied by a set of policies which constitute a policy hierarchy. Hereafter, we provide such an indicative policy hierarchy where the policies are provided as informal statements without detailed specifications for the sake of brevity. The satisfaction of this corporate goal among others assumes the following service subscription goal policy: “Ensure the responsive and reliable insertion and maintenance of the service subscribers within the videoconference service according to their SLA” (policy P1). In fact, this top-level policy is an obligation over the overall service management system, e.g., the IntegratedMan, which stands at the business management layer and must be appropriately enforced to the videoconference service provider management system (VDC_SPMan). For this policy to be satisfied, it must be appropriately refined to several sub-policies by the VDC_SPMan.

Thus, the VDC_SPMan asks from the Cloud_SPMan to: “For each new service subscriber, establish reliable connections between Videoconference Agents and the appropriate Servers” (policy P2), and from the TeleCo_SPMan (telecommunications service provision manager) to: “The telecom service provider must provide each new subscriber with committed information rate (CIR) according to the SLA’s QoS characteristics” (policy P3). Focusing further on the refinement of policy P1, the VDC_SPMan requires from the Service Subscription manager (ServSubscrMan) to: “For each new subscriber, create a usage service profile for that subscriber” (policy P4). These three policies can be considered as policies at the service management layer (Agrawal, 2021; Dursun and Üstündağ, 2021; Zhang et al., 2022).

From the telecommunications management perspective and refining P3, the TeleCo_SPMan requires from the Metropolitan Area Network Manager (MANMan) to: “For each new subscriber, select the appropriate network components for network connectivity and telecom services according to the predefined CIR” (policy P6). The MANMan requires from the RouterMan to: “For every new telecom service provision requirement, update appropriately the routing plans of Routers” (policy P8), as well as “For every new telecom service provision requirement, update the bandwidth allocation and packet priorities of Routers” (policy P9), while from SwMan to: “For every new telecom service provision requirement, update appropriately the transmission rates of the Switches” (policy P10).

Figure 9 A part of an indicative policy hierarchy for a videoconference subscription service management before (see online version for colours)



From the application service subscription management perspective and refining P4, the ServSubscrMan requires from the VCDBMan to: “For each new subscriber, install the subscriber service usage profile, as well as, the service configuration information in the respective Databases (DB’s)” (policy P7), while from the VDC agent manager (VDCAMan):“For each new subscriber, upon his connection to the videoconference network, update (deploy) the service user agent with the appropriate software release, plus any other software required within the videoconference service network (add-ons)” (policy P5). In addition, VCDBMan requires from VCDAMan to: “For each new service user agent installation and its connectivity to the database network, update the service user agent information with the DB type, drivers, versions and connectivity strings" (policy P11).

Policies deployed at the managed network elements layer can be considered as control mechanisms which is directly enforced on the managed objects such as the VDC (videoconference) agent software components, the VCDB instances, the switches, the routers, etc. (policies P12, P13, P14, P15). For example, VDC Agent Manager (VDCAMan) requires from VDC Agent to: "setup appropriately its software component parameters concerning front-end Server connectivity, DB connectivity, video compression and cryptography and access control" (policy P12).

From the above recursive refinement process, an indicative policy hierarchy has been developed as depicted in Figure 9, where policies are shown as edges of the graph while the manager objects are shown as circles. We note that the refinement of P2 is not examined here for the sake of brevity.

Please note that both IntergratedMan and VDC_SPMan concerns the layer of strategic objectives because both they assign business goal policies to the lower level managers.

6.2 Policy-based management of MAN for a videoconference service provision

In this section, we apply policy-based management on a MAN[4] (e.g., FDDI, ATM, SMDS, Ethernet-based MAN, etc.) for a videoconference service providing formal policy specifications based on Ponder-based related work (Damianou et al., 2001; Lymberopoulos et al., 2003a, 2003b; Lymberopoulos, 2004), and in our policy hierarchy development, refinement, and analysis models.

In this case study, the MAN management structure includes among others the network administrator (admin) role (the MANMan of Figures 7 and 9 belongs to this role), the security responsible (secadmin) role which belongs to the administrator's group (the SecurityMan of Figure 7 belongs to this role), and the compadmin role which is responsible for the computer videoconference agent administration (the VDCAMan of Figures 7 and 9 belongs to this role). Other roles are also defined, the instances of these roles, as well as the relationships between them. The detailed role definitions with respect to their authorisations and obligations, and the relationships between them are not provided for the sake of brevity. We note that the main goal of this case study is to provide a demonstration of policy-based management in a MAN for a videoconference application from a practical point of view, providing formal policy definitions, a respective policy hierarchy and policy hierarchy analysis examples. It is out of the scope of this paper to provide a detailed policy hierarchy deployment for the integrated management shown in Figure 7, something which is, obviously, a huge task (Agrawal, 2021; Clemm et al., 2020).

6.2.1 Authorisation, delegation of responsibility and control policies over the MAN routers

In this subsection, we focus on a set of policies concerning the authorisations and controls over the MAN routers. The positive authorisations must be accompanied by several corresponding obligations and vice versa. The admin role concerns the authenticated network administrators that are assigned with the authorisations of loading or discarding the routers from the network, as well as to activate, deactivate and control them. But as mentioned in (Alquhayz et al., 2019), the use of positive and negative authorisation policies may cause conflicts. Namely, we have the following positive authorisation policy for the admin group:

```
inst auth+ routerPolicyOp {
subject /admin/;
target /MAN/routers/;
action load(), remove(), activate(), deactivate (), control(); }
```

Furthermore, the network administrator authorises the security responsible for N, e.g., 72, hours to load the routers and activate/deactivate them between 08:00 and 24:00. After this time, the authorisation is withdrawn. This is the case of *delegation of responsibility* which is depicted by the following policy:

```
inst deleg+ (routersPolicyOp) delegroutersOp {
grantee admin/secadmin/;
target /MAN/routers/ ;
action load()->activate(), deactivate ();
if time.between("08:00","24:00")
valid time.duration(72);                }
```

Furthermore, the network administrator authorises the security responsible for the routers to deactivate after 1 hour of live meeting and time is between 08:00 and 24:00. After this period, the authorisation is withdrawn. This is the case of *delegation of responsibility* which is depicted by the following policy:

```
inst deleg+ (routersPolicyOp) delegroutersOp {
grantee admin/secadmin/;
target /MAN/routers/ ;
action deactivate ();
if time.between("08:00","24:00")
valid time.duration(1);                 }
```

The meeting host is delegated to disable video streaming if for some reason it is required (i.e., inappropriate gestures on video) as described below:

```
inst deleg+ (host) delegroutersOp {
grantee host;
target /video/ ;
action deactivate ();
if time.between("08:00","24:00")      }
```

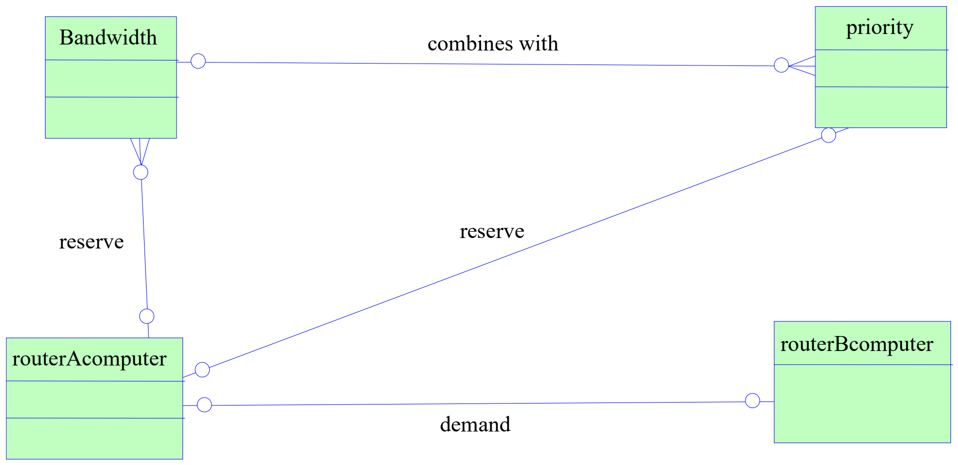
Further to these initial authorisations, if host removes a participant for any reason, the participant is kept disconnected:

```
inst oblig removeParticipant {
subject /host/;
target /participant;
action disable(participant);
in {host->meeting}
when host->remove(participant);          }
```

In addition, the host can put an attendee on hold and their video and audio connections will be disabled momentarily (see Figure 10).

```
inst oblig videoconferenceHold(BW,audio) {
subject /host;
target /participant;
on hold(participant);
do reserve(BW) && reserve(audio); }
```

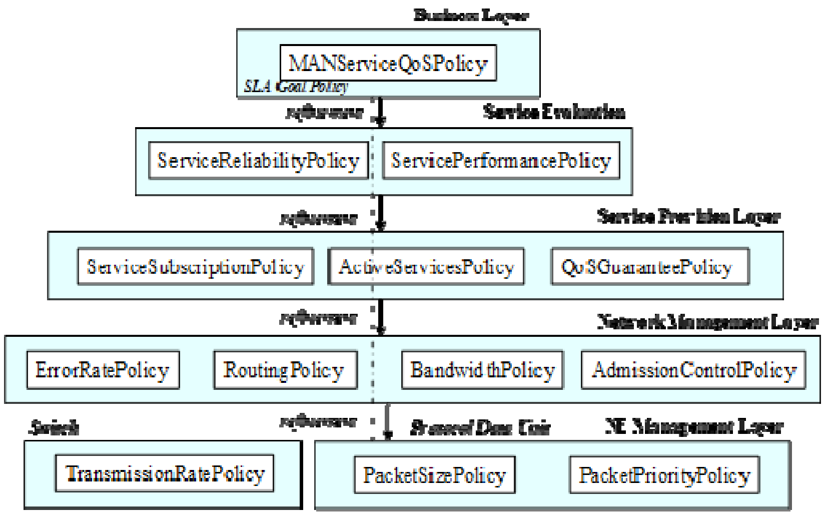
Figure 10 Videoconference request of the computers of the sub-network of router B from the computers of the sub-network of router A (see online version for colours)



6.2.2 *QoS management policy hierarchy*

The target of this sub-section is to present the construction of a rough MAN QoS policy hierarchy according to the PRF. A top-level goal policy must be first defined enforcing a repetitive policy refinement/translation process via the management structure. This refinement process starts from the top with the business-sensitive policy, named *MANServiceQoSPolicy*, which is refined to other high-level policies that mainly concern the service evaluation (SLA perspective). Service evaluation can be considered as a sub-layer of the service management layer. These policies contain information relevant to the QoS requirements for the target service and they concern the service reliability (*ServiceReliabilityPolicy*), the service performance (*ServicePerformancePolicy*), the service availability (*ServiceAvailabilityPolicy*), etc. These policies are consequently refined to the service provision policies – another sub-layer of the service management layer – that contain information which influences the QoS related behaviour of the target service, and they concern the service subscription (*ServiceSubscriptionPolicy*), the active services (*ActiveServicePolicy*), and the QoS Guarantee (*QoSGuaranteePolicy*). These policies are consequently refined to the network protocol (e.g., ATM) policies that encapsulate information that impacts the performance of the machine of the target network, and they concern the error rate (*ErrorRatePolicy*), the routing (*RoutingPolicy*), the bandwidth allocation (*BandwidthPolicy*) and the admission control (*AdmissionControlPolicy*). Finally, there are several low-level policies concerning the protocol data units (PDUs) and they contain information which influences the PDU transmission performance. These policies concern, among others, the packet size (*PacketSizePolicy*) and the packet priority (*PacketPriorityPolicy*). At the low level, the switch transmission rate policies stand, as well. It becomes clear that the various policies produced by the repetitive refinement process are in the various management layers well-known from the management of the advanced service telecommunication networks, such as UMTS, ATM, FDDI, Wi-Fi, etc. The final produced policy hierarchy for the MAN service QoS Management is shown in Figure 11. We note that for simplicity reasons the detailed relationships among the various policies are not shown, as well as to depict in a clearer way the policy allocation to the various management layers (Clemm et al., 2020; Moussa et al., 2022; Arzo et al., 2021).

Figure 11 The MAN QoS policy hierarchy according the PRF framework (see online version for colours)



6.2.3 *Policy refinement and bandwidth management*

Hereafter, we present several policies mentioned above in a formal way along with issues concerning the policy hierarchies. The policy refinement process targets to translate these policies to low (machine) level rules in order their syntax and semantics to be understood by individual devices, i.e., enforcement points (Zhao et al, 2011). Supposing that the videoconference users of a specific computer domain, e.g., of executive managers of a company, named *EM_Domain*, must have 'top performance' at the service evaluation management layer, we setup the following policy:

```
type oblig ServicePerformancePolicy {
subject /admin/compadmin/;
target r = /MAN/;
on TopVideoConferenceServiceRequest;
do r.QoSGuaranteePolicy;
when /MAN/="activated" && time.between("08:00","24:00"); }
```

Enforcing the following policy translation rule (refinement pattern) from 1 to 2 (stored in a knowledge base):

when videoconference application in computers of the *EM_Domain* (executive managers' service domain), *then*

Top service is to provide a max bandwidth and top priority in network_routers = {router1, router2, router3},

We have the following service provision policy:

```
inst oblig QoSGuaranteePolicy {
subject /admin/;
target targetSet = /MAN/routers/router1; + /MAN/routers/router2; + /MAN/routers/router3;
on Event(EventParameters(MAX_BW,MAX_PRIORITY));
do ActionParameters(bw,priority) =
CalculateActionParameters(EventParameters(MAX_BW,MAX_ PRIORITY))-
>targetSet.executeAction (ActionParameters(bw,priority));
when /MAN/routers/router1="activated" && /MAN/routers/router2="activated" &&
/MAN/routers/router3="activated" && time.between("08:00","24:00"); }
```

On *TopVideoConferenceServiceRequest* event the *ServicePerformancePolicy* is activated which consequently is refined to the *QoSGuaranteePolicy* according to translation rule which imposes the respective 'event' generation.

The above policy on routers 1, 2 and 3 is refined to network level sub-policies for setting up maximum bandwidth. Thus, the administrator is obliged to setup the routers to support a maximum bandwidth of 41 Mbps at 08:00, and a maximum bandwidth rate of 10 Mbps at 24:00 during all working days by enforcing the following obligation policy:

```
type oblig BandwidthPolicy (String timeOfDay, float MaxBW) {
subject /admin/;
target r = /MAN/routers/router1; + /MAN/routers/router2; + /MAN/routers/router3;
on timer.At(time_of_day);
";} do r.BandwidthPolicy ("08:00",41Mbps);
when time.day_of_week() != "sun" || time. day_of_week() !="sat";
do r.BandwidthPolicy ("24:00",10 Mbps);
when time.day_of_week() != "sun" || time.day_of_week() !="sat";
```

But, in case of a bandwidth upgrade, the administrator is obliged to adapt the routers to the new available bandwidth, by calculating first the new bandwidth (newBW) and generating an event via the event service for passing the new required bandwidth as a parameter. This is a request for change which must be supported from a respective change policy for instance the *BWChangePolicy* which is as follows:

```
inst oblig BWChangePolicy {
subject /admin/;
target r = /MAN/routers/router1; + /MAN/routers/router2; + /MAN/routers/router3;
on AvailableBandwidth(newBW);
do requiredBW = calculate(newBW) ->
EventService.GenerateEvent(BandwidthPolicy,requiredBW); }
```

The BWChangePolicy policy triggers another policy, the *BandwidthPolicy*, setting up its input parameters (parameterisation). In some sense, we have a *policy on policy*.

6.2.4 Packet priority policy

Upon the executive manager's demand for a videoconference session, there is also a request for top video priority. For this purpose, a policy, named 'PacketPriorityPolicy', is defined for setting up the 'priority' parameter. This policy is triggered by a higher-level policy, named 'VideoPriorityPolicy', and implemented by the 'PriorityPolicy' using the method 'calculate' and passing the new parameter value via the 'params'. The 'VideoPriorityPolicy' can be considered as a higher level produced sub-policy e.g., by the QoSGuaranteePolicy, and which is internally refined to the PacketPriorityPolicy which is a policy that affects the network behaviour. By performing the 'GenerateEvent' of the EventService the new priority is setup. In the following policy example, the initial value of priority changes from 3 to 1):

```
inst oblig VideoPriorityPolicy {
subject /admin/;
on VideoAdaptationRequest(params(priority));
do PriorityPolicy = PacketPriorityPolicy(params(priority))->
PriorityPolicy.enable(priority) ->
PriorityPolicy'sParams =
calculate (PriorityPolicy, params(newpriority)) ->
EventService.GenerateEvent (PriorityPolicy'sObligationEvent, PriorityPolicy'sParameters); }
(E. Lupu et al 1999)
```

6.2.5 MAN switch transmission rate policies

The transmission rates of MAN switches are of high interest especially for real time applications such as the videoconference that demands high volume data rates. Thus, a switch can transmit at the peak cell rate (PCR) when the transmission protocol is, e.g., ATM, the routers, 1, 2 and 3 are activated, the packet size is 512 KB, the bandwidth is at the maximum level and the network operates in the peak hours. So, we have the following policy:

inst oblig switchPCRtransmission(destin,type){
subject s = /MAN/switch/;
target t= /MAN/routers/router1; + /MAN/routers/router2; + /MAN/routers/router3;
do s.transmit(t,PCR)
when protocol =”ATM” && /MAN/routers/router1=”activated” && /MAN/routers/router2=”activated” && /MAN/routers/router3=”activated” && time.between(“08:00”, “24:00”) && packet.size=”512kb” && bandwidth=“41Mbps” }

Furthermore, the sustained cell rate (SCR) is applicable when the network operates in the peak hours, the packet size 4 MB, and for a long period (1,000 hours). So, we have the following policy:

inst oblig switchSCRtransmission(destin,type){
subject s = /MAN/switch/;
target t= /MAN/routers/router1; + /MAN/routers/router2; + /MAN/routers/router3;
do s.transmit(t,SCR)
when protocol =”ATM” && /MAN/routers/router1=”activated” && /MAN/routers/router2=”activated” && /MAN/routers/router3=”activated” && time.between(”08:00”,“24:00”) && packet.size=”4MB” && transmission.duration(1000); }

Supposing maximum bursts size (MBS), namely, the maximum number of continuously transmitted cells at PCR, using the formula: BT = ((MBS – 1) * ((1 / SCR) – (1 / PCR))), we calculate the maximum number of cells that can be submitted at PCR via the paths of routers, e.g., 1, 2 and 3 in peak hours and with packet size 512 KB. So, we have the following authorisation policy:

inst oblig switchBTtransmission(destin,type){
subject s = /MAN/switch/;
target t= /MAN/routers/router1; + /MAN/routers/router2; + /MAN/routers/router3;
do s.transmit(t,BT)
when protocol =”ATM” && /MAN/routers/router1=”activated” && /MAN/routers/router2=”activated” && /MAN/routers/router3=”activated” && time.between(”08:00”,“22:00”) && packet.size=”512kb” && BT=((MBS-1)*((1/SCR)-(1/PCR))) && burst_size=”MBS”; }

6.2.6 Policy merging and conflict detection

Supposing that whenever we have a videoconference request from the executive managers (specific user domain) through the MAN router 1, the respective bandwidth must be setup at the maximum level between 8:00 and 24:00 (policy 1), while the router1, e.g., due to release upgrade reasons, a participant must be kept out of the meeting (policy2), this causes a policy merging between the two policies related to the bandwidth allocation during the transmission of a video that must be timely detected by PHAT. Thus, setting target1 and target2 equal to router1, which means that target1 overlaps target2, and that the two actions (action1 = “remove a participant from meeting”, action2 = “kept out of operation between 18:00 and 19:00”) are merged on the same router (namely, conflict(action1, action2) is “true”) then the following Prolog statement of PHAT performs the above policy merge:

```
PolicyMerge(MAN,policy1,policy2):-
objective(policy1,target1),objective(policy2,target2),overlaps(target1,target2),
ForEvery(member(action1,policy1),member(action2,policy2), merge(action1,action2)).
```

Hereafter, the ActionConflict indicates a conflict between policies related to the bandwidth allocation during the video transmission, written in event calculus based analysis tool as follows:

```
HoldAt(ActionConflict (P1, P2, BW1, BW2), T) ß
holdsAt(Oblig(P1, Subject, operation (Targ1, setBWMax(OA1, BW1))), T) ^
holdsAt(Oblig(P2, Subject, operation (Targ2, setBWZero(OA2, BW2))), T) ^
(OA1==OA2) ^ (BW1> BW2) ^ (Targ1==Targ2 v isMemeber(Targ1, Targ2) v
isMember(Targ2, Targ1)).
```

7 Open issues and future work

Policy hierarchy creation and analysis is not an easy task. More research is needed for the resolution of several open and important issues some of which are discussed in this section. The contribution of artificial intelligence (AI), including artificial neural nets (ANN) and pattern recognition techniques can be proved very useful towards this direction. New rendering and visualisation methods could provide the administrators with supplementary support for a coherent and efficient policy hierarchical relationships analysis. Currently, middleware platforms are far behind from embedding policy hierarchy analysis services at an adequate level. In fact, the automation of the policy refinement process, especially for task-specific policies, must be further investigated and standardised. Furthermore, policy conflict detection and resolution methods need additional elaboration because the existing approaches mainly concern a few generic types of policies. In this perspective, the sub-modules of PHAT must be further examined and developed for additional policy hierarchy analysis cases to be considered. As our target is to automate the analysis process at the maximum possible extent, we have working on consistency checking between policies concerning the various ISO/ODP (Open Distributed Processing) viewpoints of a distributed system, i.e., between policies of the enterprise viewpoints with these ones of the computational viewpoint. Another research we are currently conducting is the minimisation of the overlapping between policy sets, especially for those in large-scale distributed computing environments. Policies must not be duplicated because this usually causes overheads and inconsistencies. Concluding, policy-based management provides a framework for managing organisations, applications, systems, and networks in an integrated manner rendering it a very challenging and emergency technology (Clemm et al., 2020; Moussa et al., 2022; Jiang and Wang, 2023).

Acknowledgements

The author would like to thank Professor C. Douligeris for proofreading and Mr. V. Sbyrakos (MSc) for helping in the formalisation of some policies in Ponder.

References

Agrawal, D., Giles, J., Lee, K-W. and Lobo, J. (2005a) 'Policy ratification', *Sixth IEEE International Workshop on Policies for Distributed Systems and Networks*, Stockholm, June.

Agrawal, D., Giles, J., Lee, K-W. and Lobo J. (2005b) 'Policy-based management of networked computing systems', *IEEE Communications Magazine*, October, Vol. 43, No. 10, pp.69–75.

Agrawal, N. (2021) 'Autonomic cloud computing based management and security solutions: state-of-the-art, challenges, and opportunities', *Trans. Emerging Tel. Tech.*, Vol. 32, No. 12, p.e4349, https://doi.org/10.1002/ett.4349.

Alam, A., Soltanian, A., Yangui, S., Salahuddin, M.A., Glitho, R. and Elbiaze, H. (2016) 'A cloud platform-as-a-service for multimedia conferencing service provisioning', *2016 IEEE Symposium on Computers and Communication (ISCC)*.

Alquhayz, H., Alalwan, N., Alzahrani, A.I., Al-Bayatti, A.H. and Sharif, M.S. (2019) 'Policy-based security management system for 5G heterogeneous networks', *Wireless Communications and Mobile Computing*, Hindawi, Vol. 2019, Article ID 4582391, 14pp.

Arzo, S.T., Bassoli, R., Granelli, F. and Fitzek, F.H.P. (2021) 'Multi-agent based autonomic network management architecture', *IEEE Transactions on Network and Service Management*, September, Vol. 18, No. 3, pp.3595–3618, DOI: 10.1109/TNSM.2021.3059752.

Baliosian, J. and Serrat J. (2004) 'Finite state transducers for policy evaluation and conflict resolution', *Proceedings of 5th IEEE Workshop on Policies for Distributed Systems and Networks, Policy*, IBM Watson Research Center, New York, USA, June.

Bandara, A., Lupu, E. and Russo, A. (2003) 'Using event calculus to formalise policy specification and analysis', *Proceedings of 4th IEEE Workshop on Policies for Distributed Systems and Networks (Policy 2003)*, Lake Como, Italy, June.

Bandara, A., Lupu, E., Moffett, J. and Russo, A. (2004) 'A goal-based approach to policy refinement', *Proceedings of 5th IEEE Workshop on Policies for Distributed Systems and Networks, Policy 2004*, IBM Watson Research Center, New York, USA, June.

Bandara, A., Lupu, E., Russo, A., Dulay, N., Sloman, M., Flegkas, P., Charalambides, M. and Pavlou, G. (2006) 'Policy refinement for IP differentiated services quality of service management', *IEEE ETransactions on Network and Service Management*, Vol. 3, No. 2, Second Quarter.

Bertino, E., Bonatti, P. et al (2000) 'TRBAC: a temporal role-based access control model', *5th ACM Workshop of Role-Based Access Control*, Berlin, Germany.

Bleistein, J.S., Cox, K. and Verner, J. (2006) 'Validating strategic alignment of organizational IT requirements using goal modeling and problem diagrams', *The Journal of Systems and Software*, Vol. 79, No. 3, pp.362–378, Elsevier.

Casassa, M., Baldwin, A. and Goh, C. (2000) 'POWER prototype: towards integrated policy-based management', *NOMS2000*, Hawaii, April.

Charalambides, M., Flegkas, P., Pavlou, G., Bandara, A., Lupu, E., Russo, A., Dulay, N., Sloman, M. and Rubio-Loyola, J. (2005) 'Policy conflict analysis for quality of service management', *Proceedings 6th IEEE Workshop on Policies for Distributed Systems and Networks (Policy 2005)*, Stockholm, Sweden, June.

Chen, F. and Sandhu, R.S. (1995) 'Constraints for role-based access control', *First ACM/NIST Role Based Access Control Workshop*, ACM Press, Gaithersburg, Maryland, USA.

Ciavaglia, L. and Peloso, P. (2019) *Chapter 3 A Unifying Framework Design for the Management of Autonomic Network Functions*, IGI Global.

Clemm, A., Zhani, M.F. and Boutaba, R. (2020) 'Network management 2030: operations and control of network 2030 services', *J. Netw. Syst. Manage.*, Vol. 28, No. 4, pp.721–750.

Damianou, N., Bandara, A., Sloman, M. and Lupu, E. (2002a) *A Survey of Policy Specification Approaches*, Department of Computing, Imperial College of Science Technology and Medicine, London [online] http://www.doc.ic.ac.uk/~mss/Papers/PolicySurvey.pdf (accessed 24 October 2022).

Damianou, N.N., Dulay, N., Lupu, E., Sloman, M. and Tounouchi, T. (2002b) 'Tools for domain-based policy management of distributed systems', *Proceedings of IEEE/IFIP Network Operations and Management Symposium (NOMS2002)*, Florence, Italy, April, pp.213–218.

Damianou, N., Dulay, N., Lupu, E. and Sloman, M. (2001) 'The ponder policy specification language', *Proceedings of Workshop on Policies for Distributed Systems and Networks (Policy 2001)*, January, pp.18–39, Springer-Verlag LNCS 1995, Bristol, UK.

Dohndorf, O., Kruger, J., Krumm, H., Fiehe, C., Litvina, A., Luck, I. and Stewing, F-J. (2011) 'Tool-supported refinement of high-level requirements and constraints into low-level policies', *2011 IEEE International Symposium on Policies for Distributed Systems and Networks*.

Dulay, N., Lupu, E., Sloman, M. and Damianou, N. (2001) 'A policy deployment model for the ponder language', *Proceedings of IEEE/IFIP International Symposium on Integrated Network Management (IM'2001)*, Seattle, May.

Dursun, T. and Üstündağ, B.B. (2021) 'A novel framework for policy based on-chain governance of blockchain networks', *Information Processing & Management*, Vol. 58, No. 4, Elsevier Publishers.

Forbici, E. and Penna, M. (1997) *Implementation of a RPC/IDL to CMIP/GDMO Gateway*, Tina, p.216.

Galani, A., Koutsouris, N., Tsagkaris, K., Demestichas, P., Fuentes, B., Garcia Vazquez, C. and Nguengang, G. (2012) 'A policy based framework for governing Future networks', *2012 IEEE Globecom Workshops*.

ISO/IEC (1992) *IS 10165-4: Structure of Management Information, Part 4: Guidelines for the Definition of Managed Objects* [online] https://www.iso.org/standard/18174.html (accessed 24 October 2022).

ISO/IEC JTC1/SC21 (1995) *Basic Reference Model of Open Distributed Processing, Part 2: Descriptive Model*, ITU-T X.903-ISO/IEC 10746-3.

Jiang, C. and Wang, J. (2023) 'A portfolio model with risk control policy based on deep reinforcement learning', *Mathematics*, Vol. 11, No. 1, p.19.

Kamoda, H., Yamaoka, M., Matsuda, S., Broda, K. and Sloman, M. (2005) 'Policy conflict analysis using free variable tableaux for access control in web services environments', *Proc. Policy Management for the Web, A WWW2005 Workshop 14th International World Wide Web Conference*, Chiba, Japan, May, pp.5–12.

Lamsweerde, A. (2001) 'Goal-oriented requirements engineering: a guided tour, invited minitutorial', *Proceedings of RE'01 – 5th International Symposium in Requirements Engineering*, Toronto, August, pp.249–263.

Letier, E. and Lamsweerde, A. (2004) 'Reasoning about partial goal satisfaction for requirements and design engineering', *Proceedings of SIGSOFT'04/FSE-12*, ACM, Newport Beach, CA, 31 October–6 November.

Lupu, E. and Sloman, M. (1999) 'Conflicts in policy-based distributed systems management', *IEEE Transactions on Software Engineering*, Vol. 25, No. 6, pp.852–869.

Lupu, E., Milosevic, Z. and Sloman, M. (1999) 'Use of roles and policies for specifying and managing a virtual enterprise', *Proceedings of 9th IEEE International Workshop on Research Issues on Data Engineering: Information Technology for Virtual Enterprise (RIDE-VE'99)*, Syndey, Australia, March.

Lymberopoulos, L. (2004) *An Adaptive Policy based Framework for Network Management*, PhD thesis, Department of Computing, Imperial College, University of London, October.

Lymberopoulos, L., Lupu, E. and Sloman, M. (2003a) 'An adaptive policy based framework for network services management', *Journal of Network and Systems Management, Special Issue on Policy Based Management of Networks and Services*, Vol. 11, No. 3, pp.277–303.

Lymberopoulos, L., Lupu, E. and Sloman, M. (2003b) 'Using CIM to realize policy validation within the ponder framework', *DMTF Gobal Management Conference*, San-Jose, California, June.

Lytras, M.D., Visvizi, A., Chopdar, P.K., Sarirete, A. and Alhalabi, W. (2021) 'Information management in smart cities: turning end users' views into multi-item scale development, validation, and policy-making recommendations', *International Journal of Information Management*, Vol. 56 No. C.

Miller, A. and Dess, G. (1996) *Strategic Management*, 2nd ed., McGraw-Hill, London.

Mitropoulos, S. (2000) 'Integrated enterprise networking management: case study in intelligent multimedia message handling systems', *Journal of Network and Systems Management*, Vol. 8, No. 2, pp.267–297.

Mitropoulos, S. (2021) 'An integrated model for formulation, alignment, execution and evaluation of business and IT strategies', *Int. J. Business and Systems Research*, Vol. 15, No. 1, pp.90–1111, Inderscience Publishers.

Mitropoulos, S. (2022) 'Conflict resolution in management policies: case study in cloud computing and internet of things', *International Journal of Information Privacy, Security and Integrity*, Vol. 5, No. 2, pp.93–110, Inderscience Publishers.

Mitropoulos, S. and Douligeris, C. (2006) 'A policy-driven methodology for managing telecommunication networks', *Annual Review of Communications 2006*, Vol. 59, IEC Publications, USA.

Mitropoulos, S. and Douligeris, C. (2010) 'Integrated policy-based governance of virtual enterprises and systems', to appear to the *International Journal of Systemic Studies (IJASS)*, Vol. 3, No. 3, pp.326–342, Inderscience Publishers.

Mitropoulos, S. and Veldkamp, W. (1994) 'Integrated distributed management in interconnected LANs', *Proceedings of IEEE/IFIP Network Operations and Management Symposium (NOMS' 94)*, Florida, USA, February, pp.898–908.

Moffett, J. and Sloman, M. (1993) 'Policy hierarchies for distributed systems management', *IEEE JSAC Special Issue on Management*, Vol. 11, No. 1, pp.1404–1414.

Moore, B., Ellesson, E. et al. (2001) *Policy Core Information Model – Version 1 Specification*, Network Working Group – RFC3060 [online] http://www.ietf.org/rfc/rfc3060.txt (accessed 24 October 2022).

Moussa, A., Mounir, K. and Pierre, R. (2022) 'A review of IoT network management: current status and perspectives', *Journal of King Saud University – Computer and Information Sciences*, Vol. 34, No. 7, pp.4163–4176.

OASIS (2001) *OASIS eXtensible Access Control Markup Language (XACML) TC* [online] https://www.oasisopen.org/committees/tc_home.php?wg_abbrev=xacml (accessed 24 October 2022).

Radwan, N., Abdelhalim, M.B. and AbdelRaouf, A. (2019) 'Implement 3D video call using cloud computing infrastructure', *Ain Shams Engineering Journal*, June 2020, Vol. 11, No. 2, pp.363–375.

Ravuri, H.K., Vega, M.T., van der Hooft, J. et al. (2022) 'A scalable hierarchically distributed architecture for next-generation applications', *Journal of Network System Management*, Vol. 30, No. 1, p.1, Springer.

Ribeiro, C., Zuquete, A. et al. (2001) 'SPL: an access control language for security policies with complex constraints', *Network and Distributed System Security Symposium (NDSS'01)*, San Diego, California.

Rubio-Loyola, J., Serrat, J., Charalambides, M., Flegkas, P. and Pavlou, G. (2006) 'A methodological approach toward the refinement problem in policy-based management systems', *IEEE Communications Magazine*, October, Vol. 44, No. 10, pp.60–68.

Sloman, S. (1994) 'Policy driven management for distributed systems', *Journal of Network and Systems Management*, Vol. 2, No. 4, pp.333–360.

Sloman, S., Magee, J., Twidle, K. and Kramer, J. (1994) 'An architecture for managing distributed systems', *Proceedings of 4th IEEE Workshop on Future Trends of Distributed Computing Systems*, Lisbon, September.

Vinoski, S. (1997) 'CORBA: integrating diverse applications within distributed heterogeneous environments', *IEEE Communications Magazine*, February.

W3C (2006) *W3C Member Submission, Web Services Policy 1.2 Framework* [online] http://www.w3.org/Submission/WS-Policy/ (accessed 24 October 2022).

Zhang, J., Lu, N., Ma, J. et al. (2022) 'A secure access control framework for cloud management', *Mobile Netw. Appl.*, Vol. 27, No. 1, pp.404–416, Springer.

Zhao, H., Lobo, J., Roy, A. and Bellovin, S.M. (2011) 'Policy refinement of network services for MANETs', *Proceedings of the 12th IFIP/IEEE International Symposium on Integrated Network Management (IM 2011)*, Dublin, Ireland, 23–27 May.

Notes

1 http://www.webopedia.com/TERM/V/videoconferencing.html (accessed 24 October 2022).

2 https://www.sciencedirect.com/science/article/pii/S2090447919301297 (accessed 24 October 2022).

3 http://www.iec.org/online/tutorials/tmn/ (accessed 24 October 2022).

4 http://www.iec.org/online/tutorials/tmn/ (accessed 24 October 2022).