Proceedings of the IFIP/IEEE International Symposium on Integrated Network Management,
Santa Barbara, California, USA, 1 - 5 May 1995

Using a Classification of Management Policies for Policy Specification and Policy Transformation

René Wies
Munich Network Management Team
University of Munich, Department of Computer Science
Leopoldstr. 11b, 80802 Munich, Germany
Phone: +49-89-2180-3139
Email: wies@informatik.uni-muenchen.de

Abstract

Policies are derived from management goals and define the desired behavior of distributed heterogeneous systems, applications, and networks. To apply and deal with this idea, a number of concepts have been defined. Numerous policy definitions, policy hierarchies and policy models have evolved which are all very different, as they were developed from diverse points of view and without a common policy classification.

This paper presents and structures the characteristics of policies by introducing a general classification for policies and showing how this classification leads to and aids in the specification of policies. Furthermore, we outline the ideas of a policy life cycle, and that of policy transformation. Policy transformation is a refinement process with conflict resolution which converts policies to become applicable within a management system using management services, such as systems management functions, distributed services, etc.

The paper further looks at aspects to be considered when defining policy templates and concludes with a number of open issues still to be looked at in this field of management policies.

Keyword Codes: K.6.4; C.2.4
Keywords: Network and Systems Management, Management Policy, Policy Classification, Policy Transformation, Policy Hierarchy, Policy Templates

1 Introduction and Motivation

The primary and overriding objective of network and systems management is to maintain network and system availability and aid in extending the network and systems, enhance the performance, provide security, reduce operating overhead (repetitive tasks), and decrease the cost of running the information technology infrastructure. Despite the fact, that providers of network and system services are aware of these objectives, the problem of translating these goals into actions remains. A possible approach to tackle this problem are management policies, which provide a (semi-) formal concept to record and structure the objectives, to refine them depending on the infrastructure, and apply them through the use of management systems.

Policies, as we define them, are derived from management goals and *define the desired behavior of distributed heterogeneous systems, applications, and networks*. It is important to recognize that policies specify only the *information* aspects of this desired behavior, i.e. *what* behavior is desired; they do not describe the precise actions to be taken, i.e. *how* the behaviour can be achieved and maintained.

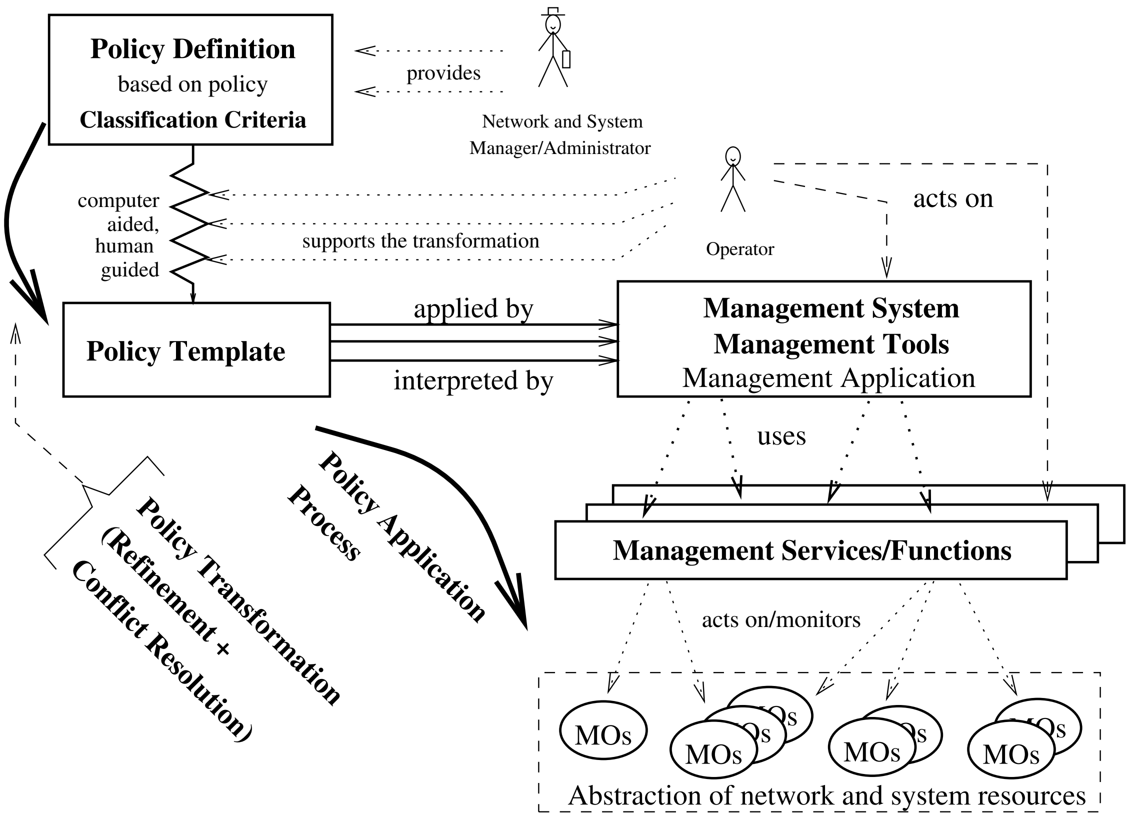

Figure 1: Transformation and application of policies

Low level, technical policies may be wrongly seen similar to the `behaviour` attribute in GDMO ([ISO 10165-4]) templates for managed object classes (MOCs). However, whereas the behavior template for use in a managed object classes (MOCs) defines the *possible or available* behavior of the resource it represents, a policy defines the *desired* behavior, i.e. it is a restriction on the possible behavior. For high level policies this analogy cannot be drawn and we will deal with the policy hierarchy in Section 3.1.

In contrast to the ongoing standardizing work on domains and policies ([ISO 10040/2], [ISO 10164-19], [ISO 10746-1]), our definition indicates that policies are primarily independent from the concept of domains, yet policies can be either applied to or used to define domains of managed objects.

As will be described in Section 3.1, policies may range from high level i.e. abstract non-technical policies to low level technical policies, depending on how the desired behavior of the managed resources is specified. However, unlike [MACA 93] we do not see policies to cover the wide spectrum from business goals and strategies (societal and economic policies) to the executable policies (procedural policies), nor are our policies necessarily executable by some *unsophisticated program* as suggested in [BEHO 93]. The level of abstraction in terms of the desired behavior of distributed heterogeneous systems, applications, and networks, depends on the degree of detail contained in the policy definition and the *ratio* of business related aspects to technological aspects within the policy (see Figure 4 and Section 3 for a detailed discussion). Thus, policies as we will deal with them for the remainder of this paper, do not describe business goals but are derived from them, nor are they executable management scripts, even though management scripts could be generated from low level policies ([WIES 94]).

Only once we know exactly what aspects characterize policies and how policies can be processed, we can start to embed the concept of *management policies* into a suitable management architecture, or possibly extend existing or develop new management architectures. It is our goal to combine for example the numerous formal concepts for the definition of technical security policies (e.g. [MARR 93], [WARE 94]) and the abstract architectures for the application of business or corporate policies (e.g. [IDSM 93]) into *one comprehensive concept* which can deal with policies of all levels in the hierarchy. Furthermore, to avoid the task of having to define implementation specific extensions as is the case for existing Managed Object definitions [HABO 91], the structure and components of a policy object definition must take their future realization (implementation and application) into account. On the grounds of the issues presented throughout the following sections, we will briefly discuss examples of commercial systems and network management tools near the end of the paper in Section 4.2.

Figure 1 illustrates the main ideas described in this paper. The classification presents valuable input for both, the definition of new policies and the realization of policies from existing policy catalogues. The classification criteria are the aspects to be exactly looked at when defining policies. The transformation process is the most difficult part, as it must convert generally abstract and informal policies into low level policies which can be applied to the environment. The transformation process primarily consists of refinement steps and possibly some conflict resolution ([MOFF 94]). *The end products of this transformation process are not policies that act directly on managed resources but rather specifications on how to apply management tools and how to utilize management functions or management services offered by a management system.* Besides the concepts of policy classification, transformation, and application, two other concepts, policy hierarchy and policy life cycle, are introduced to complete the picture of management policies. Yet, the policy classification builds a common basis for all following issues, as it summarizes and organizes the important characteristics of policies.

2 Policy Classification

The large number of policies calls for a classification, i.e. a well-defined set of (as far as possible orthogonal) grouping criteria. The main goal of such a classification of policies is:

1. to get a better grasp of what is meant by management policies and what can be achieved through their use.

Other goals are for example:

2. to identify differences and commonalities between policies in order to specify different classes of policies;

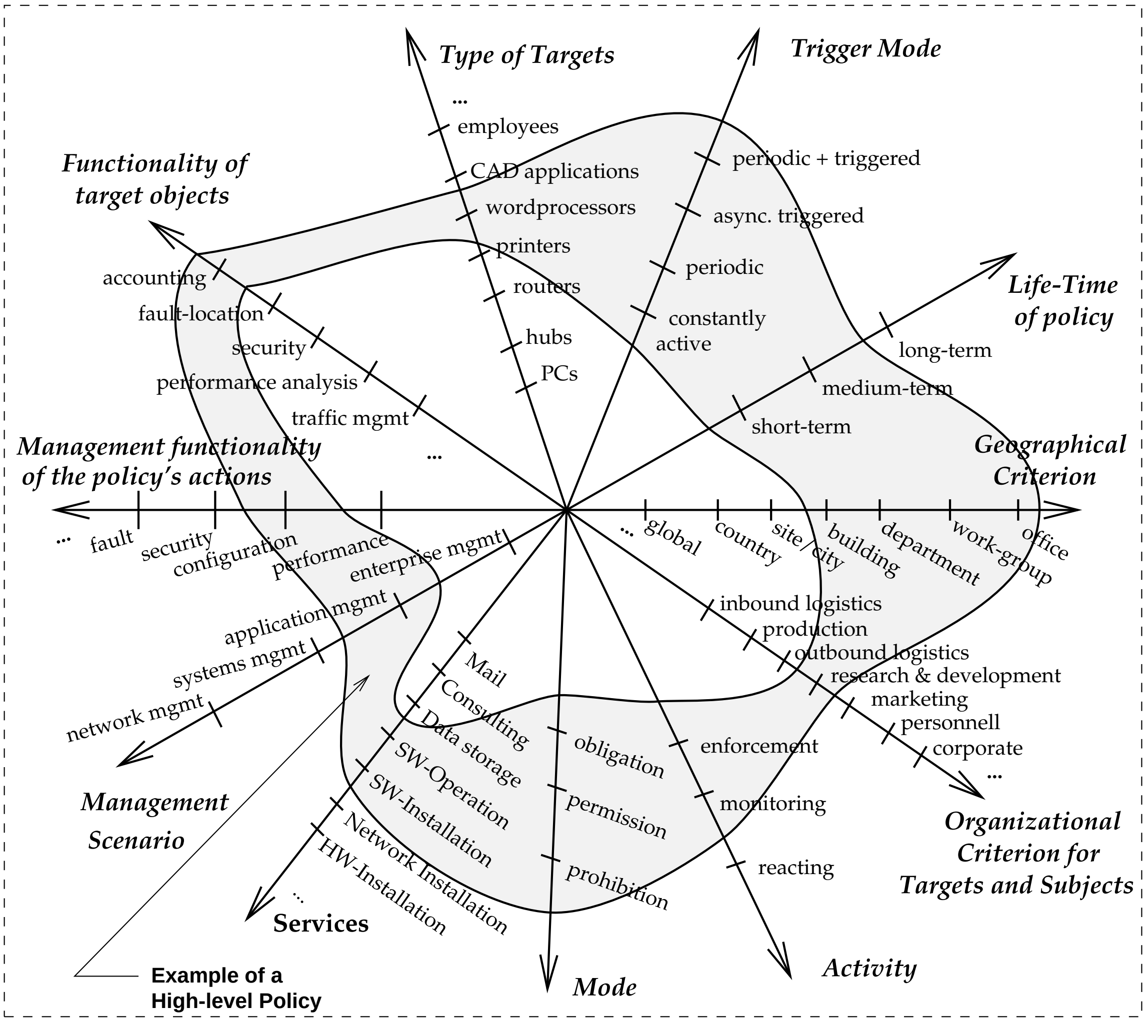

Figure 2: Criteria for Policy Classification

3. to derive a policy hierarchy for the process of policy transformation; and
4. to derive and verify the components of a formal definition of policies.

Several network and system service providers (e.g. FidoNet, VirNet) have gathered their policies in *policy catalogues* which are written in an informal way. A structure, if at all present, is given by the services the company offers to its customers, e.g. policies specific to mail services, data storage, data processing, consulting services, software installation. A thorough analysis of these policy catalogues from numerous network and system service providers and talks with network and system managers, administrators, and operators (e.g. at debis, BMW, LRZ) have allowed us to collect a list of criteria for the classification of policies which are illustrated in Figure 2 in form of a multi-dimensional diagram.

Most of these dimensions can be associated with one or more of the ODP viewpoints[1]

[1] A list of the dimensions and their associated ODP viewpoints would be beyond the scope of this paper, especially as this association is very vague and hence of little use.

([ISO 10746-1]). However, using only the five ODP viewpoints would again group different characteristic properties of policies together and would thus not help to explain and organize management policies.

The precise labels of the axes, i.e. the different categories for each criterion inevitably depend on the level of abstraction, i.e. the policy's position in the policy hierarchy, which will be discussed later. However, the *names* of the dimensions (e.g. trigger mode, life time) will remain the same, whichever level of abstraction we look at; only the labels are refined within each dimension.

For the sake of brevity, we will not explain the different dimensions further, as most of them are self-explanatory. In this Figure, a high level policy (drawn in light grey) covering several categories per dimension is indicated. It describes a real-life example for the following scenario: *The computer science faculty consists of several departments, each of which owns the same number of floating licenses for the word processing system as there are full-time researchers at each department. There are also a limited number of spare licenses for part-time researchers which are distributed on demand. The policy, stating that the departmental licenses must be used before the spare licenses are allocated and distributed, must be enforced and its enforcement monitored*. Using the above classification can only supply a very simplified representation of the policy, as the refinement of the categories and dimensions as well as the notion of domains are neglected. However, this example illustrates, that every dimension must be considered when defining a policy.

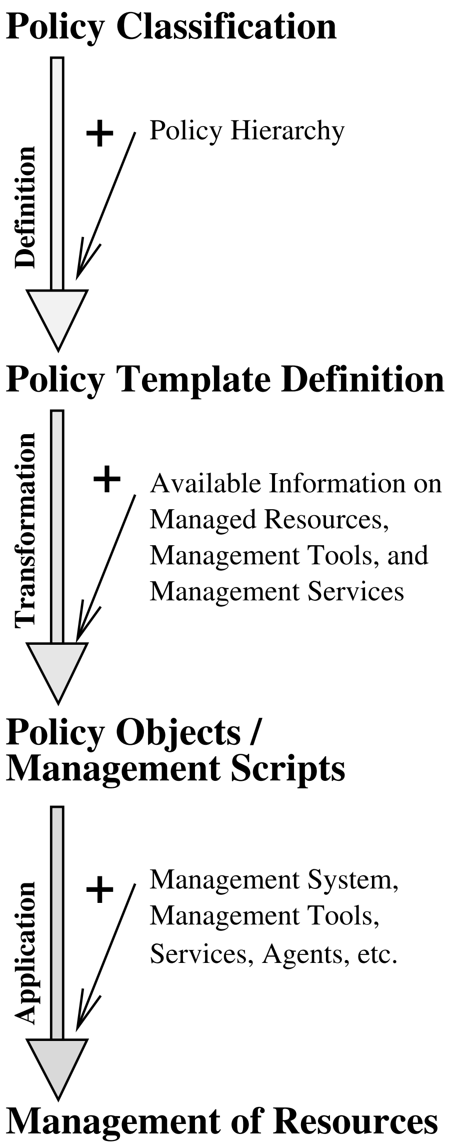

Figure 3: The simplified path from policy classification to policy application

As the above stated goals show, this classification is not designed to describe a policy completely, or even to cover all aspects of a policy. For example the axis *type of targets, functionality of targets, geographical criterion* and *organizational criterion* may appear as *one* attribute called *target domain* in a policy template. These domains may either be resolved during the transformation process or remain as is in the policy template to be resolved later by some other domain-resolver in the management system. This issue will be discussed further in Section 3.2 when we take a look at the transformation process.

The classification provides a basis for the derivation and structuring of policies as well as possible hints towards their transformation. In addition, the refinement of the axes in combination with the application of a policy hierarchy will lead to one policy template definition suitable for all levels of the hierarchy. A policy, no matter from which level of the hierarchy, can be analyzed and structured along the above criteria. This process of defining, analyzing, and structuring policies is the starting point for the processing and application of management policies. This approach is illustrated in Figure 3.

3 Policy Hierarchy and Transformation Process

3.1 Policy Hierarchy

When analyzing catalogues of policies from various network and system service providers, it becomes apparent how different policies can be. Different in terms of their values for the above classification dimensions and in their level of abstraction or degree of detail. Security policies specifying the precise format of the allowed password structure or the IP addresses of systems to be protected by firewalls are *mixed* with abstract policies describing the required availability and accessability of printers or policies documenting the precautions to be taken when using a specific management tool.

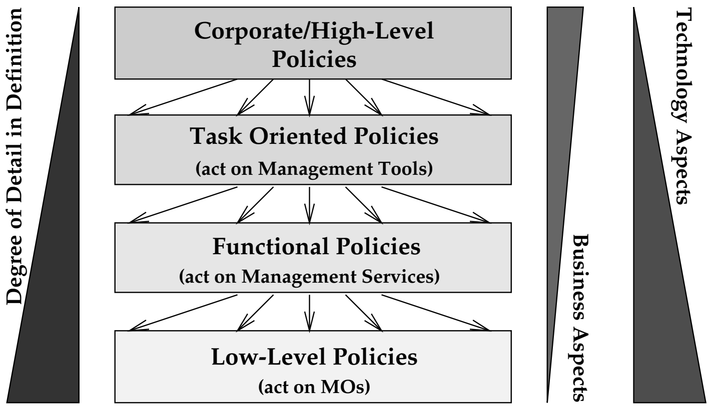

Figure 4: The Policy Hierarchy

To guarantee that all policies are applied to their targets (provided they are not in conflict with each other), it is essential to structure these policies. Thus, a policy hierarchy is a way of splitting the vast number of policies into smaller groups of different levels of abstraction, which can be further processed in distinct steps and transformed into applicable low-level policies. Examples of policy hierarchies can also be found in [MACA 93] and [NGUY 93].

The levels of the hierarchy also represent different *views* on policies. Examples of such views are: the view of a corporate network manager who only sees and only specifies corporate/high level policies; or the view of a network operator, who sees functional policies and realizes them through the use of a management system which in turn may use specific management functions or management services.

Thus, a policy hierarchy defines the levels within the management environment at which policies are applied. As Figure 4 illustrates, the policy hierarchy distinguishes between the following:

- Corporate policies or high level policies: These are directly derived from corporate goals and thus embody aspects of strategic business management rather than aspects of technology oriented management. To allow their application within the management environment, they have to be refined to one of the three policy types below.
- Task oriented policies: Their field of action is sometimes referred to as task or process management, where they define the way how management tools are to be applied and used to achieve the desired behavior of the resources.
- Functional policies: These policies operate at the level of and define the usage of management functions, such as the OSI *systems management functions* ([ISO 10164-X]), the OSF/DME *distributed services* ([DME 92]), or OMG's *object services* ([OMG 92a, OMG 92b]); and
- Low level policies: They operate at the level of managed objects (MOs). MOs in this context refer to simple abstractions of managed network and system resources, and not MOs for e.g. systems management functions.

If a policy can be implemented by use of management functions or management services the last level of the hierarchy may not be reached during the refinement process, nor may it be necessary to define policies at this low level. Furthermore, certain policies can be assigned to exactly one level of the hierarchy, yet other (less well defined) policies may be assigned to different levels and thus must be split into separate policies before the transformation process can be applied.

3.2 Transformation Process

Following the definition of policies using the above classification, each characteristic property can be further detailed to allow a stepwise refinement of the policy. In other words, the lower the level of abstraction, the more precise and detailed will the definition become, i.e. the granularity of the criteria increases. However, in addition to the refinement of a policy, the transformation process can also be used to identify the targets, subjects, and necessary monitor objects. For example, a high level policy calling for a weekly backup of all the company's data may be refined to specify the backup media for different workstation clusters and also identify the system administrators, that are responsible for operating stackers or changing tapes.

To illustrate the refinement of policies, the axis labeled *management functionality* for example can be further subdivided to describe the policy's actions more precisely. As illustrated in Figure 5, an unspecific high-level security policy may be further refined into two separate policies, one responsible for assuring the confidentiality of data and the other for assuring the confidentiality of traffic flow ([ISO 7498-2]).

The number of stages in this transformation process may vary depending on the axes and their labels. Thus, for some axes a derived value may not be refined further while for other dimensions the process of refinement must carry on until the final value is determined.

The questions that arise now are:

- How is this transformation achieved ?

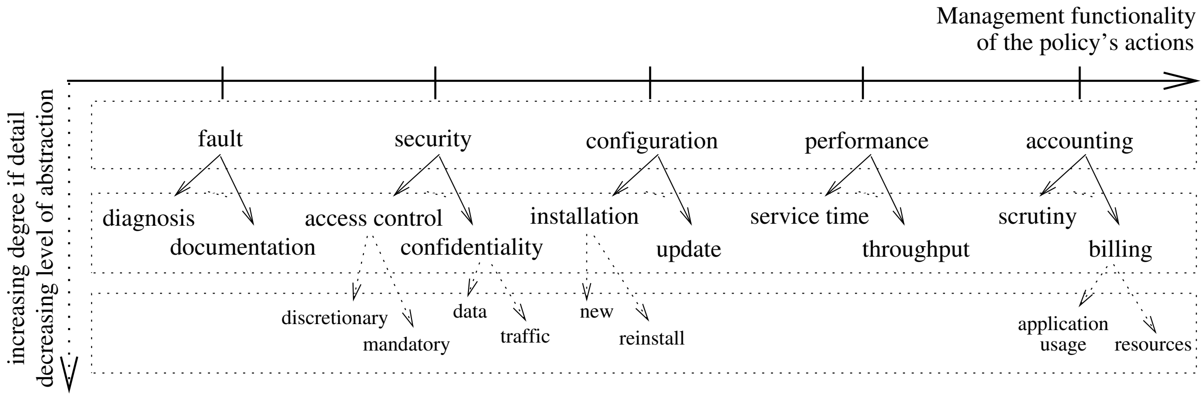

Figure 5: Refinement of classification criteria

- When does this process of refinement end ?

To answer the first question, in some cases this process may be automated, yet generally we expect to apply the idea of *computer aided - intuition guided* processing [BBBD 85], i.e. with the helping hand of an expert operator. The question whether this transformation process can be automated or to which degree an automation can be achieved cannot be answered at this stage. However, to interpret the semantics of policies and for any automation of this process (fully computerized or human guided), extensive management information on the managed environment, the management capabilities of the involved systems, and information on available tools, platforms, etc., is essential.

A completely different approach, could be to limit this transformation process to a syntactical transformation which could make concepts like *skolem reduction* applicable. Yet, neglecting semantical interdependencies of policies is not satisfactory as these will probably cause the majority of conflicts.

The transformation will end, when the reached degree of detail cannot be refined further or when a mapping between the value (object, action, etc.) to managed objects or management functions of the management system is possible. Thus, it is a process of merging the results from a *top-down* approach (i.e. the refinement of policies) with the results from a *bottom-up* approach (i.e. the analysis of available management functionality). For example, if the derived targets or monitor objects can be related to existing MOs or if the management actions to be performed can be mapped to management functions or services, the process of refinement will end. However, if a transformation is not possible, for whichever reason (lack of information, conflicts, etc.) the policy may need to be re-defined or taken care of by a human operator. This process was summarized in Figure 1.

The example of floating licenses introduced in Section 2 would need to be refined for example to identify the license servers which are to be configured or to identify the clients which need to be monitored to verify the policy's enforcement. Furthermore, the trigger mode (asynchronous triggering by license requests) and life time of the policy would need to be further detailed during the transformation process.

In Section 2 we already mentioned the problem of resolving domains. This can be either done during the transformation process or resolved later by a domain-resolver within the management system. The latter approach has the advantage of a more simple policy transformation process but allows no (or very limited) conflict resolution until the policy is actually applied. It merely shifts the complexity of resolving conflics from the transformation process to the management system which applies the policy. The former approach (conflict resolution during transformation) leads to a more complex transformation process with e.g. backtracking methods, but it also causes severe problems when it comes to dynamically changing domain-members. For example, newly added devices/objects to a target domain must be dynamically added to the policy's targets, which may result in new conflicts, possibly new monitoring strategies, or even a complete new transformation of the policies concerned. However, to deal with or even answer the question of which alternative for conflict resolution is more practical and sensible, is beyond the scope of this paper.

3.3 Policy Life Cycle

Before we move on to the derivation of policy templates, the policy life cycle is introduced, as it provides valuable information on aspects to be incorporated in template definitions. The life cycle does not only influence the attributes of a policy template (e.g. trigger mode, lifetime), it also hints towards the actions and notifications to be specified within a template and the management functions and services required to implement a policy. The life cycle is characterized by the fact that a policy can divided into an enforcement and a monitoring part ([WIES 94]). The policy enforcement part can be activated by trigger objects (asynchronous events) or through the use of monitor objects.

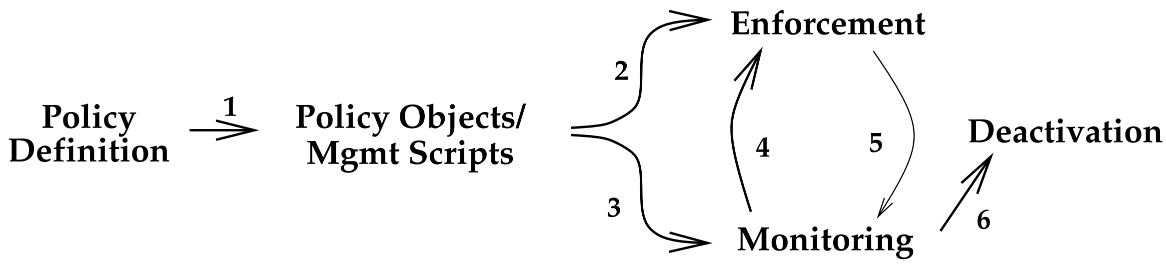

Figure 6: The policy life cycle

The numbers in Figure 6 are to be interpreted as follows:

1. policy transformation: as described in Section 3.2, high level policies are refined and transformed into low-level policies and further processed to become applicable within the management environment.
2. application of *active* policies: policies are activated through the management system or specific management applications. *Active* policies first carry out certain actions and later monitor changes upon which the policy may again react, provided such actions are specified in the policy. (Suspending and resuming policies will be treated as part of policy enforcement.)

3. application of *monitoring* policies: *Monitoring* policies have no initial enforcement part and only monitor certain MOs and possibly react if necessary. The monitoring may also be done using for example a monitoring systems management function.
4. policy adaption or change: The reaction upon changes during the lifetime of a policy can be treated just like the initial enforcement actions. This is because changes in the managed environment may lead to a change in the overall enforcement of the policy, for example additions to a target domain may require a completely new configuration of all other domain members.
5. changes leading to new requirements on monitoring, triggering or enforcement actions: As for the above situation, a change in the enforcement actions may require a new monitoring strategy. For example the deletion of one domain member may no longer require the monitoring of this resource.
6. deletion of policies: short and medium term policies will become obsolete at some point in time. For example when they are replaced by new policies or their domain of targets is removed from the environment.

From the above life cycle, certain characteristics concerning the functionality of necessary underlying management services can be derived. For example, a policy must be able to emit notifications concerning the change in a target's characteristics, or actions must be carried out on the policy if a domain is changed. Furthermore, functions to activate, pause, resume, delete or change a policy must be specified to allow an effective implementation and application of policies.

4 Policy Templates and Commercial Products

4.1 Derivation of Policy Templates

In this section we will use the term policy template rather than policy object, because the notion of objects tends to call for a formal definition of objects which can be implemented using some object oriented language, compiled with some sophisticated compiler, and simply applied to the environment. However, research in this field is far from this stage, even though standards are already being defined.

A policy template must be defined for policies of all levels of the hierarchy. Thus we need a template which suits abstract high level policies as well as technical low level policies. Furthermore, the syntax must allow an effective and efficient use by both managers and operators, for specifying and refining policies. In [SLOM 93] a policy structure consisting of five object attributes (modality, subject scope, target scope, activity, and constraints) is proposed. This may be sufficient for a very abstract representation of policies that involve human operators, but even for high level and abstract technical policies, more information must be held in the template in a structured format. Based on the classification criteria of Figure 2, a policy template could have the format shown in Figure 7. However, not all dimensions are represented by components in the proposed template because for example, the modality is of little use when trying to refine policies or apply them to management services.

```
POLICY TEMPLATE
    Author(s):
    CreationDate: (mm/dd/yy)
    StatusOfRefinement: (pending, completed/applicable, stopped
        due to conflicts, stopped due to lack of information, etc.)
    DerivedFromParentPolicy:
    GoalAndActivity: (free-text, detailed and semi-formal description
        of what is to be enforced and monitored;
        and how to react to changes)
    ManagementScenario: (network management, systems management,
        application management, enterprise managementt)
    ManagementFunctionality: (fault, accounting, configuration,
        performance, security management)
    Service: (services involved or effected by the policy)
    LifeTime: (duration of application)
    SubjectCharacteristics/Domain: (tools, mgmt. functions, etc.)
    TargetCharacteristics/Domain: (functionality, site, type, etc.)
    TriggerMode: (async.Triggered, syncronous, asyncMonitoring,
        periodicMonitoring, etc.)
    TriggerCharacteristics/Domain: (monitoring objects, triggering events, etc.)
    PolicyProcessOrScript: (formal description of the management
        script or management process/steps to be executed to
        enforce the policy)
    Notifications: (emitted notifications due to policy
        violations, enforcement/monitoring failures, etc.)
REGISTERED AS { ... }
```

Figure 7: Example of a policy template

Actions, or methods as they are called in the object oriented world, could be grouped into a set of administrative actions for creating, deleting, pausing, resuming a policy application, and sets of operational actions for example for adding or deleting targets, subjects or monitors; or sets of actions for changing other characteristics of a policy such as the trigger mode or time mode. The ultimate goal is of course the mapping of these templates to management functions and services as discussed earlier.

4.2 The Scope of Commercial Products

Almost all major vendors of management systems and platforms ([JAND 94]) have recognized the advantages of using policies and domains in integrated management. Products such as HP's Dolphin ([PGMM 93]), Cabletron's MaestroVision ([MAES 93]), and Tivoli's TME offer some rudimentary functionality to define domains and apply policies. Yet, all concepts are proprietary and very limited in their functionality.

For example, the systems management system HP-Dolphin uses an object-oriented Prologlike language for the specification of policies The transformation of a policy is not automated nor system supported, but must be done manually by the programmer. Furthermore, these low level policies are applied to system-specific, non-generic, and non-standardized objects.

Tivoli's Management Environment ([WELL 94]) defines *Core Object Services* and *Common Management Services*, including services for the definition of *Policy Regions* (commonly known as domains) and *Policy Objects*. These Policy Objects consist of a set of customizable programs usually written as shell or Perl scripts. They are not designed to use other management services to enforce policies nor is the concept open for the integration of global and system-independent policies.

5 Conclusions and Future Work

Policies are a powerful concept for the management of distributed heterogeneous systems, networks and applications. In this paper we did not present finished work, but rather ongoing research.

Our classification criteria and the template generated from them have proven to allow a fairly complete description of policies and their characteristics. Thus, new policies can be defined and existing *raw* policies may be detailed using the classification criteria. The refinement process followed by mapping the enforcement and monitoring activities to management functions can be done manually, provided of course management functions are available at the lowest level. Yet, the automation, or at least a systematical approach, supported by a *policy specification and application tool* to achieve this refinement is essential to apply (large numbers of) policies. Therefore, the refinement process marks the main focus of our future research in this field. Thus, while our definition, concepts and classification criteria appear to be powerful as well as natural, in our next step we will prove that they are also practical by refining policies from different management scenarios. However, the goal is not to design a fully automated "policy-system" but rather to develop a structured approach for a generally manual but computer aided, tool supported transformation and application of management policies.

Acknowledgements

The author wishes to thank the members of the Munich Network Management Team for fruitful discussions and valuable comments to preliminary versions of this paper. The MNM Team directed by Prof. Dr. Heinz–Gerd Hegering is a group of researchers of the University of Munich, the Technical University of Munich, and the Leibniz Supercomputing Center of the Bavarian Academy of Sciences.

References

[BBBD 85] F.L. Bauer, R. Berghammer, M. Broy, W. Dosch, F. Geiselbrechtinger, R. Gnatz, E. Hangel, W. Hesse and B. Krieg-Brückner, *The Munich Project CIP, Vol 1: The Wide Spectrum Language CIP-L.*, volume 183 of *Lecture Notes in Computer Science*, Springer, 1985.

[BEHO 93] Karsten Becker and David Holden, “Specifying the Dynamic Behavior of Management Systems”, In Manu Malek, editor, *Journal of Network and Systems Management*, volume 1, pages 281 – 298, Plenum Publishing Corporation, September 1993.

[DME 92] Open Software Foundation, *OSF Distributed Management Environment (DME) Architecture*, 1992.

[DSOM 91] IFIP, *Proceedings of the IFIP/IEEE International Workshop on Distributed Systems: Operations & Management*, October 1991.

[DSOM 93] IFIP, *Proceedings of the IFIP/IEEE International Workshop on Distributed Systems: Operations & Management*, October 1993.

[HABO 91] H.-G. Hegering, S. Abeck and Th. Böhnke, “Converting MIB-Descriptions into MIB-Implementations”, In [DSOM 91].

[IDSM 93] “Domain and Policy Service Specification”, IDSM Deliverable D6 / SysMan Deliverable MA2V2, IDSM Project (ESPRIT III EP 6311) and SysMan Project (ESPRIT III EP 7026), October 1993.

[ISO 10040/2] “Information Technology – Open Systems Interconnection – Systems Management Overview – Amendment 2: Management Domains Architecture”, PDAM 10040/2, ISO/IEC, November 1992.

[ISO 10164-19] “Information Technology – Open Systems Interconnection – Systems Management – Part 19: Management Domain and Management Policy Management Function”, CD 10164-19, ISO/IEC, January 1994.

[ISO 10164-X] “Information Technology – Open Systems Interconnection – Systems Management – Management Functions”, IS 10164-X, ISO/IEC.

[ISO 10165-4] “Information Technology – Open Systems Interconnection – Structure of Management Information – Part 4: Guidelines for the Definition of Managed Objects”, IS 10165-4, ISO/IEC, August 1991.

[ISO 10746-1] “Basic Reference Model of Open Distributed Processing – Part 1: Overview and Guide to Use”, WD 10746-1, ISO/IEC, November 1993.

[ISO 7498-2] “Information Processing Systems – Open Systems Interconnection – Basic Reference Model – Part 2: Security Architecture”, IS 7498-2, ISO/IEC, 1988.

[IWSM-I 93] Wesley W. Chu and Allan Finkel, editors, *Proceedings of the IEEE First International Workshop On Systems Management, Los Angeles*, IEEE, April 1993.

[JAND 94] Mary Jander, “Management Frameworks”, *Data Communications International*, February 1994.

[MACA 93] M. Masullo and S. Calo, “Policy Management: An Architecture and Approach”, In [IWSM-I 93].

[MAES 93] Calypso Software Systems, “MaestroVision 2.0 beta 1”, Release Notes, Calypso Software Systems, Inc., 1993.

[MARR 93] Randy Marchany, "Writing a Site Security Policy: RFC 1244", In [IWSM-I 93].

[MOFF 94] Jonathan D. Moffett, *Specification of Management Policies and Discretionary Access Control*, chapter 17, pages 455–481, In [SLOM 94], June 1994.

[NGUY 93] Thang Nguyen, "Linking Business Strategies and IT Operations for Systems Management Problem Solving", In [IWSM-I 93].

[OMG 92a] "Object Management Architecture Guide", Document 92-11-1, Object Management Group, September 1992.

[OMG 92b] "Object Services Architecture", Document 92-8-4, Object Management Group, August 1992.

[PGMM 93] Adrian Pell, Chen Goh, Paul Mellor, Jean-Jacques Moreau and Simon Towers, "Data + Understanding = Management", In [IWSM-I 93].

[SLOM 93] Morris Sloman, "Specifying Policy for Management of Distributed Systems", In [DSOM 93].

[SLOM 94] Morris Sloman, *Network and Distributed Systems Management*, Addison-Wesley, June 1994.

[WARE 94] Willis H. Ware, "Policy Considerations for Data Networks", *Computing Systems, The USENIX Association*, 7(1):1–44, 1994.

[WELL 94] Caroline Wells, "Tivoli Systems, Inc., Tivoli Management Environment (TME)", *Datapro Integrated Network Management*, January 1994.

[WIES 94] René Wies, "Policies in Network and Systems Management – Formal Definition and Architecture –", In Manu Malek, editor, *Journal of Network and Systems Management*, volume 2, pages 63 – 83, Plenum Publishing Corporation, March 1994.

Biography

René Wies received his diploma (Diplom-Informatiker, M.Sc.) in computer science from the Technical University of Munich, Germany, and a MBA-MDP degree from the Graduate School of Management, Boston University in Japan. Currently he is a Ph.D. student at the University of Munich and a member of the Munich Network Management Team, directed by Prof. Dr. Heinz-Gerd Hegering. He does research on integrated network and systems management, emphasizing on management policies. He is a member of the IEEE and GI.