# Policy-Based Management for Call Control

**Kenneth J. Turner**
*University of Stirling, UK*

## INTRODUCTION[1]

This article discusses the use of policies to control calls—whether in traditional telephony or in its more modern versions such as mobile telephony or Internet telephony. Call control is as old as telephony. It allows subscribers and the network to manage calls. Trivially, users initiate calls by dialing and terminate calls by hanging up. However, modern telephony offers many more options for managing calls. For example, they may be forwarded if the user is busy or away. Conference calls may be set up. Voicemail and answering services can be used to take messages.

The solutions in conventional telephony are, however, relatively limited. This article investigates the relevance to call control of the kinds of policies used to manage computerized and networked systems. As will be seen, policy-based management of call control offers a much more flexible approach.

## BACKGROUND

### Call Control

In telephony, the basic call is extended through features. These are relatively self-contained additions of functionality, for example, for call diversion, call waiting, or charge card calling. An important aspect of features is that they are automatically invoked, usually at well-defined trigger points in the basic call state model. This means that features can readily be added with little disturbance to the basic call. Unfortunately, the same mechanism means that features may interfere with each other—the well-known feature interaction problem (Cameron et al., 1993).

A policy is a high-level statement of what actions are permitted under what conditions. For example, an organization might define a policy that urgent calls should be forwarded if not answered within five seconds. Policies conflict if they dictate inconsistent actions, for example, a call should be both rejected and forwarded. There is a good analogy between call features and call policies, and between feature interaction and policy conflict. In a sense, a feature is a low-level policy. However, features have limited flexibility (e.g., low-level nature, restricted parameterization, defined by the network operator). In contrast, policies are higher-level and more malleable. There are similarities between features and policies, but also important differences (Dini et al., 2004; Reiff-Marganiec & Turner, 2004). Features and policies are both intended to allow users to control their calls. Feature interaction and policy conflict may be handled statically (at definition time) and also dynamically (at call time).

However, features are low-level and imperative, whereas policies are higher-level and declarative. Suppose the user does not wish to receive calls from the press. In a feature-based approach, terminating call screening would be required with a list of blocked numbers. A comparable policy would simply reject calls from the press, identified by the caller domain or the topic.

Features have limited parameters, whereas policies can be much more flexible. For example, a call diversion feature would typically be parameterized by the affected number, the forwarding number, and the condition for diversion. This is as far as conventional features can be customized. A comparable policy could be much more subtle, choosing different forwarding numbers according to the caller, the time of day, the subject of the call, the capabilities and devices of the call parties, and so forth.

Features are fixed and managed by the network operator or equipment supplier, whereas policies are open-ended and defined (mostly) by end users. A typical network or switch may have tens to hundreds of features. Although this may offer the user many options, the range of choices is nonetheless fixed. If the user's requirement is not met by an existing feature, there is little alternative. Because features are defined by engineers, a technically complex approach may be followed. In contrast, policies should be definable by users to meet their needs. Although a policy language necessarily limits what users may do, the range of policies is much wider and is in fact unlimited. Since policies should be accessible to ordinary users, a user-friendly and non-technical approach must be adopted.

Despite standards for signaling, feature interaction handling is essentially under the control of one network operator or equipment supplier. This makes it much easier to identify and manage feature interactions. Policies, however, may be user-defined. Furthermore, the policies applying to a call may stem from any pair of users (who may have never called each other before). Detecting and resolving conflicts among such policies is thus a much more complex and dynamic task.

## Policy-Based Approaches

Policy-based management has become popular for controlling a variety of systems. As examples, policies are commonly used for access control, quality of service, and system management. Policies capture high-level goals that can be automatically enforced. Using predefined policies, a system can dynamically adjust its behavior without requiring manual intervention.

Policies have been used in many kinds of management tasks. Example applications include access control (Belokosztolszki & Moody, 2002), admission control (Yavatkar, Pendarakis, & Guerin, 2000), agent-based systems (Buhr et al., 1998), content distribution (Verma, Calo, & Amiri, 2002), distributed trust (Seamons et al., 2002), group collaboration (Pearlman, Welch, Foster, & Kesselman, 2002), healthcare (Aljareh & Rossiter, 2001), network management (Marriott, Mansouri-Samani, & Sloman, 1994), open distributed processing (Steen & Derrick, 1999), quality of service (Ponnappan, Yang, & Pillai, 2002), security (Ryutov & Neuman, 2002), and systems management (Damianou, Dulay, Lupu, & Sloman, 2000).

Lupu and Sloman (1999) define policies as information that can be used to modify the behavior of a system. This is a very general and open-ended definition. In the context of this article, policies are the goals for how calls should be handled. Policies lend themselves well to networked applications, where the very distribution demands careful management. Despite this, call-handling systems have attracted little policy support. Amer, Karmouch, Gray, and Mankovskii (2000) use fuzzy policies as a means of resolving feature interactions. Many researchers see policies as important in future call handling (Dini et al., 2004).

Policy language developments in industry have largely focused on network management and QoS (quality of service). For example, Cisco has developed policy support for control of security and QoS in routers. Lucent and Bell Labs developed PDL (Policy Description Language) for network management. Hewlett-Packard's PolicyXpert (now discontinued) was also focused on network management. The IETF standard for COPS (common open policy service) is intended as a protocol for managing QoS. None of these efforts is of direct relevance to call control.

Policy conflict is an almost inevitable consequence of policy-based management. Such conflicts may arise at different levels. The policies of one user may interfere with each other. Someone needing a network printer, for example, may have high quality and low cost as goals. The policies of peer users may also disagree. For example one user in a videoconference might desire high quality video, while the other requires low quality due to limited device capabilities. Policies may also be defined hierarchically within an organization. Conflicting policies may occur at all levels, for example, individual (high-quality video needed), department (H.261 video codec preferred), organization (video bandwidth should be limited).

## Policies for Call Control

CPL, the Call Processing Language (Lennox & Schulzrinne, 2001), allows users to define how they wish calls to be handled. However, CPL is limited in a number of ways that make it unsuitable for general call control. CPL is limited in its network bindings. It also gives limited control over calls, specifically just call setup. There is a need for mid-call control (e.g., when a new party is added to a call) and call tear-down control (i.e., when a call is disconnected). CPL does not offer capabilities found in other systems such as the use of preferences, integration with presence and availability, and handling of conflicts among user preferences. Some of the limitations of CPL have been addressed in work on LESS: Language for End System Services (Wu & Schulzrinne, 2003). New developments in this include support for presence-based services and consideration of feature interactions.

Call centers and CTI (computer telephony integration) support flexible call handling (see Gans, Koole, and Mandelbaum, 2002, for a survey of the approaches). Call centers rely on mechanisms such as calling line