Managing the Enriched Experience Network – Learning-Outcome Approach to the Experimental Design Life-Cycle

Samson Lee, Nigel Sheridan-Smith, Tim O'Neill, John Leaney, Kumbesan Sandrasegaran and Shmuel Markovits

***Abstract*— Experimental design methods have long been used in scientific areas such as agriculture, biology and physics to minimise error and assure validity. Although most network researchers performing experiments with testbeds and simulations implicitly follow scientific method, until recently there has been little emphasis on improving experimental design methods. Traditional experimental design focuses on experiments where the scope and objectives are relatively constrained, however network research in innovative areas where there is little or no precedence often has changing objectives that evolve over time. We describe the learning-outcome approach to the experimental design life-cycle that applies the concepts of systems development life-cycle models used in software engineering, as well as learning taxonomies used in education. Our approach extends traditional experimental design by providing a more comprehensive and efficient way of decomposing an experimental research project into manageable stages that are designed rather than improvised, leading to a well-structured way of assessing the knowledge gained. We provide an insight into our experiences with this approach in the context of experimental research in management of Alcatel Australia's new-generation Enriched Experience Network.**

***Index Terms*— experimental research, experimental design, network management, enriched experience network, learning taxonomy, systems development life-cycle**

I. Introduction

THE problem space for the *management* of Alcatel Australia's new-generation *Enriched Experience Network* (EEN)[1] is very broad, making *experimental research* through simulation and testbed environments a daunting task. Traditionally, *experimental design* (ED) techniques have been used as a strategy for planning, conducting, analysing and interpreting experiments so that sound and valid conclusions can be drawn efficiently, effectively and economically. ED provides experimenters a greater understanding and power over the experimental process. However, the evolving objectives particularly during the early stages of our project, the number of factors associated with complex distributed systems, and the multi-disciplinary nature of our research effort requires an approach that not only encompasses ED as a part of the *systems development life-cycle* (SDLC) methodology, but explicitly takes into account the *learning outcomes* at each stage of the project.

The contribution of this paper is the reporting of our early experiences with planning the *experimental design life-cycle* (ED life-cycle) in the context of experimental research in the management of Alcatel Australia's new-generation EEN. To our knowledge, there have been no published papers about applying ED as a part of the SDLC. To our knowledge, there have been no published papers about using the learning taxonomies in the context of this paper.

The remainder of this paper is organised in four sections. In Section II, we provide references to some publications about ED. In Section III, we describe the ED life-cycle as a part of the SDLC. In Section IV, we describe how two common learning taxonomies can be used to better understand the learning outcomes at each stage of the project.

II. Experimental Design

R.A. Fisher [1] pioneered a major portion of theory within the subject of ED in the 1930s, and the statistical methods have matured in the fields of agriculture, health science and physics. Hinkelmann and Kempthorne describe the three principles of *replication*, *randomisation* and *blocking* (local control) that need to be considered in the design of experiments through *treatment design*, *error-control design* and *sampling and observation design* in [2].

The topic of ED applied to simulations can be found in a number of papers. Hatami et al [3] describes experiences of designed experiments as part of their simulation project methodology from a manufacturing perspective. Work by Donohue [4] provides an overview of research on strategic design issues that are unique to experimentation in a simulation environment. Blosch and Antony [5] provide a case study of the Royal Navy's use of ED and computer-based simulation. Law and Kelton [6] describe methods to help design the runs for simulation models and interpreting their output.

While their work focuses on designing individual experiments, they do not consider the process of breaking up a large research project into multiple stages in the ED life-cycle.

Submitted to the 2003 Australian Telecommunications, Networks and Applications Conference (ATNAC'03) on 31 August 2003; Accepted 17 September 2003; Revised 17 October 2003; Presented at ATNAC'03 Melbourne, Australia 8-10 December 2003.

This work was partially supported by the UTS-Alcatel Australia Linkage Grant for the Architecture-based Engineering Research Program at the Institute for Information and Communication Technologies, UTS.

S. Lee is with the Faculty of Engineering, University of Technology, Sydney, Australia (e-mail: samson.lee@uts.edu.au)

[1]The commonly used term, Next Generation Network (NGN) has become quite diluted in meaning, with some organisations using it simply to mean everything IP, or the use of converged infrastructure for voice, video and data. Alcatel Australia's new-generation EEN transcends the classical view of NGN by highlighting *Quality of Experience* (QoE) in the value proposition.

III. The Experimental Design Life-Cycle

Some experiments can be planned with all objectives known at the beginning of the project, while larger and more complex experiments can only be designed after discoveries are made from earlier exploratory experiments. When the problem space is very broad, it is logical to break up the experiments into stages that are more manageable. Instead of simply changing the objectives as discoveries are made, the experiments need to systematically *evolve* over time. Consideration must be given to ensure that the experiments are designed and not just improvised whenever changes to objectives are made. Our approach to ED recognises that experimental research has a life-cycle with a beginning, an operational life and a conclusion similar to the SDLC.

Thiele introduces the basic concepts of SDLC processes in [7]. An SDLC model depicts the significant phases or activities of a software systems project from conception until the product is retired. It specifies the relationships between project phases, including transition criteria, feedback mechanisms, milestones, baselines, reviews, and deliverables.

Three fundamental life-cycle models are introduced in ISO/IEC TR 15271 [8] and further refined using system life-cycle stages in ISO/IEC 15288 [9]. They are the waterfall, incremental, and evolutionary models. Typically, a life-cycle model addresses the following phases of a software project: requirements, design, implementation, integration, testing, operations and maintenance.

In the context of experimental research, we are designing and performing experiments instead of designing and implementing software systems. The requirements phase in the SDLC model is adapted to the objective, questions and hypotheses phase of the ED life-cycle. The software design phase adapts to the ED life-cycle considerations consisting of *treatment design*, *error control design* and *observation design* as described in [2]. The implementation phase adapts to actually performing the experiments to obtain results. Instead of resulting in a software build that is integrated, tested and used, we result in knowledge that is synthesised at the conclusion of each stage of the ED life-cycle.

The choice of which model to use in the ED life-cycle depends on a number of things: how well known the objectives, questions and hypotheses are; the number of experimental factors that have been identified; how well the investigator understands the problem (as classified in the learning outcome taxonomy in the next section); whether the objectives are expected to change and evolve throughout the project; and how soon the investigator is required to produce documentation to communicate progress status.

The *waterfall model* is essentially a single-pass approach based on experimental objectives, questions and hypotheses gathered at the start of the project. The single pass approach is similar to the classical ED process for an individual experiment, and has relevance to projects with relatively simple and well-defined experimental objectives. This model is impractical for most research projects because it is unable to deal with the complexity of numerous factors. The remaining two models are more suitable for complex experimental research.

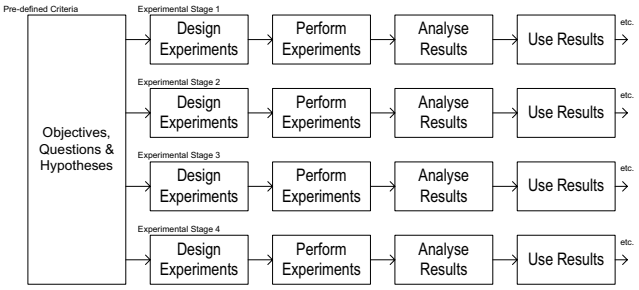

Fig. 1. Incremental Experimental Design Life-Cycle Model

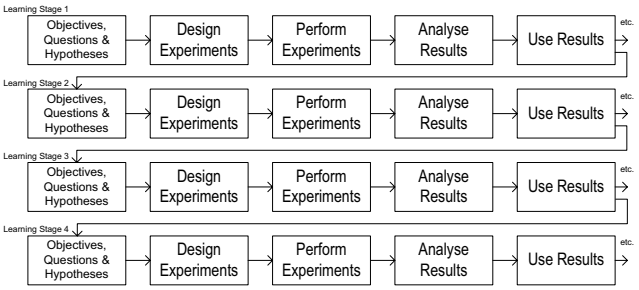

Fig. 2. Evolutionary Experimental Design Life-Cycle Model

The *incremental model* (Fig. 1) uses a pre-defined set of multiple passes based on a set of experimental objectives, questions and hypotheses defined at the start of the project but that are implemented through a series of stages. The objectives are not expected to change throughout the project. The stages may be serial and / or partially overlapped in parallel. Though the objectives, questions and hypotheses are all known and defined at the start of the project, the relationships between each individual experiment does not need to be known. All of the factors that can be controlled are known before performing any experiment. This model is similar to factor grouping in traditional ED. Experience gained at each finished stage does not significantly affect the objectives of the next stage. From the researcher's point of view, the knowledge and understanding of the topic will increase incrementally.

The *evolutionary model* (Fig. 2) differs in that the experimental objectives, questions and hypotheses are progressively derived and refined with each stage. This is ideal in cases where objectives are unclear or can not be defined due to lack of knowledge in the area being investigated. It is important with this model to constrain the number of stages to an appropriate number as the initial lack of clear experimental objectives will often result in some back-tracking. This model is best suited when there has been no precedence to the research being performed. Experience with each finished stage is incorporated in objectives for the next experimental stage. From the researcher's point of view, the knowledge and understanding of the topic will "evolve" over time.

In practice, these fundamental models may be combined to create hybrid models more suitable for a particular situation. For example, the investigator may begin with a number of simple experiments using an incremental approach. This is useful because the results of these simple experiments may be documented early in the project to demonstrate the issues to

management and non-technical personnel. Following the first set of simple experiments, a number of stages may be planned using an evolutionary approach.

Much of the motivation behind utilising a life-cycle model is to provide structure to avoid the problems of the 'undisciplined hacker'. In the same manner, encompassing ED as part of the SDLC methodology allows investigators to break up a large experimental project into manageable stages – an approach that might be familiar to some engineers, but not all. However there are some key differences to the design of systems and the design of experiments for research. The ultimate goal of systems design is to make a product that satisfies requirements, whereas in research, the investigator is interested in gaining knowledge and understanding a topic. Developed systems are more tangible and foreseeable than the knowledge that is uncovered in research. This learning process needs to be formally characterised so that we are actively aware of the outcomes and capabilities at each stage of the ED life-cycle.

IV. The Learning-Outcome Approach

The application of learning taxonomies such as Bloom's *Cognitive Taxonomy of Learning*[2] [10] and Biggs and Collis' *Structure of the Observed Learning Outcome (SOLO)*[3] [11], [12], [13] can help in understanding the learning outcomes at each stage of the project. By approaching the ED life-cycle with a focus on learning outcomes, these taxonomies allow the project goals and objectives to evolve without compromising design, and provides a convenient structure for management to keep track of progress in terms of knowledge gained. In a multi-disciplinary research effort such as ours, the learning-outcome approach helps plan the ED life-cycle by structuring the understanding of each other's perspectives in order to integrate separate ideas into a coherent whole.

Some examples of how SOLO can be used to characterise levels of performance when carrying out experimental research are provided below.

At the first level, termed *pre-structural*, the investigator is unable to or can not design an experiment or test a hypothesis. This is the stage where either nothing can be done or something is done for no real reason other than the fact that it can be done. This is the stage that describes the "undisciplined hacker", who might be skilled at connecting wires together, but is unable to have *a priori* questions and hypotheses. There is nothing wrong with starting out by exploring the problem space at a pre-structural level because this experience is often the stepping stone to the subsequent levels.

In the second level, termed *uni-structural*, the investigator designs single experiments and provides simple hypotheses. The ED assumes that the answer can be found in one step, and that there is only one answer. For example, a student researching Differentiated Services might provide a single hypothesis that traffic marked as Expedited Forwarding will be protected even when co-existing with heavy background traffic. In this case, the student is not too sure why this happens and is not aware of cases where this might not be true.

In the third level, termed *multi-structural*, the investigator designs more than one experiment and provides multiple hypotheses but generally does not inter-relate these. The ED still tends to assume that the answer can be found in one step, but a collection of single step experiments are offered to choose from. For example, the student in the previous example might provide another hypothesis that traffic marked as Expedited Forwarding will start to be dropped when the throughput exceeds a certain rate. In this case, the student is not too sure why this happens and is not aware of the effects of dropping aggregate traffic.

In the fourth level, termed *relational*, the investigator designs more than one experiment, provides multiple hypotheses, and demonstrates a clear understanding of inter-relationships between these – either through some reasoned logic, a theory or through a method of evaluating and selecting the most appropriate design and hypothesis. ED is characterised by a multi-stage approach to a determination of the "facts". For example, the student in the previous examples now also has a clear understanding that Expedited Forwarding traffic will only be protected if the PHB has been over-provisioned, and that if much aggregate traffic is dropped, then there will be adverse effects for numerous micro-flows.

In the fifth level, termed *extended* abstract, the investigator designs more than one experiment and provides multiple hypotheses that not only demonstrate a clear understanding of interrelationships but also extend the initial problem. For example, the student in the previous examples recognises some issues with Differentiated Services and extends the problem by suggesting there may be a need for admission control management to solve this particular issue.

The application of the SOLO taxonomy in experimental network research is one way of structuring the knowledge gained from each experimental stage. Depending on the difficulty of the experimental research, it may be beneficial to begin by constraining the objectives, questions and hypotheses to the early levels in order to obtain familiarity and gain a complete understanding of the problem space. As the understanding of the topic increases, this naturally leads to the progression to more advanced levels.

V. Experiences at Alcatel Australia

Our project is framed around the problem of managing Alcatel Australia's new-generation Enriched Experience Network (EEN). One aspect is Policy-based Network Management, which is briefly described in the first part of this section.

To perform experimental research and to verify and validate our results, we are developing the Open Experimental Platform (OEP) consisting of a number of advanced network testbeds and simulation environments that are representative of core, edge, access and customer layers of EENs. We are highlighting the importance of the learning-outcome approach to the ED

[2]Bloom created this taxonomy for categorizing level of abstraction of questions that commonly occur in educational settings. The six stages in increasing complexity are: knowledge, comprehension, application, analysis, synthesis, and evaluation.

[3]Biggs provides a systematic way of organising how a learner's performance grows in complexity when mastering many tasks. The five stages in increasing complexity are: pre-structural, uni-structural, multi-structural, relational, and extended abstract.

life-cycle in the OEP to minimise error, assure validity and increase efficiency. We describe our experiences with selecting stages in the ED life-cycle for our research in the second part of this section.

A. What is Policy-Based Network Management?

Policy is a generalised concept that formalises the specification of system behaviour. A particular application of this concept is called Policy-based Network Management (PBNM). This is a novel approach that fuses IP networks with the service-level management concepts that are more common in traditional telecommunications network management systems.

Most IP networks are managed in a device-centric fashion since there are numerous issues with the Simple Network Management Protocol (SNMP) [14], [15], [16]. Since SNMP has not kept pace with the evolution of IP network devices and each vendor's specific features, it is primarily used for periodic monitoring. Devices tend to be configured manually by Command Line Interfaces (CLI) or semi-automated scripts. The lack of a coherent and service-orientated management platform for IP networks is a serious issue for EENs that will significantly increase operational costs and lower availability as operators struggle to keep up with their unreliable and unpredictable networks.

PBNM can potentially contribute to a number of important areas in network management. Providing a cohesive interface that performs comprehensive network management functions helps operators to cope with the increasing complexity of the network through a unified information model [14]. This interface also provides greater opportunities for automation [17] and inter-operability [18] since vendor-specifics can be hidden by increasing the abstraction of the configuration models. Large-scale networks will benefit the most since the scalability of management will be vastly improved and service-level management will become the primary concern of these automated network management systems.

EENs go far beyond the requirements of the existing IP networks that are prevalent in enterprises since they must deal with carrier-class issues. The introduction of multiple diverse services that will be customer and service focussed establishes the need for measures that provide a Quality of Experience (QoE) to the subscribers to ensure their satisfaction and enjoyment of the services that they are paying for [16]. This translates to some capability for end-to-end Quality of Service (QoS) within the network itself, since application performance is a large contributor to contentment with the offered services.

Unfortunately, end-to-end QoS is still not a reality and comprehensive and reactive management systems are necessary to close the gap on service management through intelligent Traffic Engineering [19] and other approaches. This is often called *dynamic policy*, since the behaviour is modified over time. One form of dynamic policy is *feedback*, where monitoring activities by the management system lead to changes in the low-level policies [14], [17], [20], [21]. In this case, the management system is adjusting the network according to changes in demand or changing network conditions. Another form of dynamic policy is *outsourcing* the management responsibilities to other agents and the network devices to respond to dynamic events [22].

It is well noted that policies are to be expected to change dynamically due to changing network conditions or due to *evolution* of needs and requirements over time. In this case policy is a means of making management systems programmable without a full system re-development [23]. This flexible and graceful evolution will help to improve adaptability and reduce management system churn.

This is not to say that the policy road is without ordeals. The only policy schema mapping currently defined is LDAP-based, which lacks change notification and transactional/referential integrity [14], [20], [24] that are necessary for dynamic policy management. Additional areas of work include non-QoS models, policy language standardisation, policy refinement and conflict detection/resolution.

It is important to recognise that few of these claims about policy have ever been validated formally or applied to real-life production systems of large scale [17]. Regardless of whether "policy" is used or not, we must evaluate the importance of changes to the configuration of the network to maintaining service-level qualities. Here lies the importance of the learning-outcome approach to the ED life-cycle in its application to experimental research and validation of these policy claims.

B. Stage Selection

Network management and policy are becoming more complicated: this requires us to start somewhere simple to build up a more complex understanding of what policy is and where it will help. We consider that the differences between having static and dynamic policies are significant to network management, since one does not perform any configuration changes, whereas the other modifies the static policies in accordance with monitoring and feedback from the network. Once we make observations in simple experiments, we can evolve our ED to formulate new objectives and hypotheses according to the results. There might be a need to apply more iteration or recursion if the results largely contradict our expectations.

Additionally we consider that policy becomes more complex through its pervasiveness in two dimensions. The horizontal dimension considers the richness of the policy model in capturing the mechanisms for control and monitoring. The vertical dimension considers the levels of abstraction of the policy models that help to bridge the gap between high-level requirements and SLAs and low-level device behaviours. These additional dimensions are other ways that we can expand the complexity and breadth of our experiments in terms of analysing policy. The horizontal dimension provides more adaptability, whereas the vertical dimension adds more dynamic behaviour and responsiveness.

We will start with experiments to examine traffic behaviour without QoS provisions (no policy control - "best-effort" traffic handling, no policing, etc). This will be followed by the introduction of simple, static policies such as the application of

TABLE I
POSSIBLE STAGES

Static and Dynamic	Horizontal Pervasiveness	Vertical Pervasiveness
No Policy	No Policy	No Policy
Static Policy	Centralised Policy	Device-Level Policy
Dynamic Policy	Partially Distributed Policy	Medium-Level Policy
Architected Policy	Distributed Policy	Service-Level Policy

Differentiated Services, queuing and scheduling, traffic policing, etc. By keeping the configuration relatively unchanged through the experiment we can analyse and evaluate the behaviour of the network under a fixed configuration. We can see the influence that the treatment has on the network traffic and apply ED approaches to reduce the likelihood of error. This will provide us with insight that helps us to understand the benefits and limitations of various techniques for QoS.

Through this approach we are testing our understanding within our limited scenarios and incrementally validating the results that we obtain.

The next step is to formulate and develop models of how dynamic QoS policy can be applied to influence network and traffic behaviour. Ideally, the management system will use feedback from monitoring and network measurement to influence its decisions about how to control the network configuration to improve efficiency, manage service quality, or to deal with changes in the network environment. We can test hypotheses that we set for ourselves to evaluate whether the actions we have specified improve or worsen the situation according to our expectations. Ideas here can be gathered by examining one or more existing approaches used in research systems and comparing them quantitatively. A basic proposal for our evolutionary stages is shown in Table 1.

VI. CONCLUSION AND FUTURE WORK

Our early experiences with planning the ED life-cycle for the management of Alcatel Australia's new-generation Enriched Experience Network has been very positive. The learning-outcome approach has been found to be very effective and efficient for assessing the progress of the project, particularly because of the broad objectives and multi-disciplinary nature of the research effort.

We highlighted the importance of methodology in network research and presented the framework to a learning approach to the ED life-cycle. In essence, to efficiently perform ED in our experimental research, the concepts from systems development life-cycles and the learning-outcome taxonomies can be applied. The application of software life-cycle management is advantageous since research projects using testbeds and simulations are often complex in nature requiring rigourous and well-structured methodology. Since network research is essentially a learning process, the application of the SOLO (learning outcome) taxonomy helps to structure the knowledge gained from each stage of the ED life-cycle.

We are currently using and further assessing this approach and will be documenting our experiences with the learning-outcome approach to the ED life-cycle. A number of detailed experiments have been planned.

ACKNOWLEDGMENT

The authors wish to thank Mark Hunter from Alcatel Australia for his support in the project. Thanks to Professor Peter Miller and Dr. Deirdre Cobbin from the Department of Health Sciences at the University of Technology, Sydney for their help on experimental research methodology.

REFERENCES

[1] R. A. Fisher, *The Design of Experiments*. Edinburgh: Oliver & Boyd, 6th ed., 1951.
[2] K. Hinkelmann and O. Kempthorne, *Design and Analysis of Experiments Volume I - Introduction to Experimental Design*, vol. 1. John Wiley & Sons, 1994.
[3] S. Hatami, E. Cowley, and C. Morey, "Using experimental design to improve the efficiency of simulation modeling-a manufacturing perspective," in *Simulation Conference, 1990. Proceedings., Winter*, pp. 310–313, 1990. TY - CONF.
[4] J. Donohue, "Experimental designs for simulation," in *Simulation Conference Proceedings, 1994. Winter*, pp. 200–206, 1994. TY - CONF.
[5] M. Blosch and F. Antony, "Experimental design and computer-based simulation: a case study with the royal navy," *Managing Service Quality*, vol. 9, no. 5, pp. 311–319, 1999.
[6] A. M. Law and W. D. Kelton, *Simulation Modeling and Analysis*. McGraw-Hill Higher Education, 3rd ed., 2000.
[7] D. Thiele, "Life cycle management - using life cycle process standards," *The journal of Software Engineering Australia*, 2002.
[8] ISO/IEC-TR-15271, "Information technology – guide for ISO/IEC 12207 (software life cycle processes)," 1998.
[9] ISO/IEC-15288, "Systems engineering – system life cycle processes," 2002.
[10] B. Bloom, *Taxonomy of educational objectives: The classification of educational goals: Handbook I, cognitive domain*. New York ; Toronto: Longmans, Green, 1956.
[11] J. B. Biggs and K. F. Collis, *Evaluating the Quality of Learning - The SOLO Taxonomy (Structure of the Observed Learning Outcome)*. Academic Press, 1982.
[12] J. B. Biggs, "Assessing for learning: Some dimensions underlying new approaches to educational assessment," *Alberta Journal of Educational Research*, vol. 41, no. 1, pp. 1–17, 1995.
[13] D. Nulty, "Characterisation of levels of performance in terms of the SOLO taxonomy," 2000.
[14] M. L. Stevens and W. J. Weiss, "Policy-based management for IP networks," *Bell Labs Technical Journal*, pp. 75–94, 1999.
[15] J. Saperia and J. Schonwalder, "Policy-based enhancements to the SNMP framework," tech. rep., May 2000.
[16] T. Hamada, P. Czezowski, and T. Chujo, "Policy-based management for enterprise and carrier ip networking," *Fujitsu Science and Technology*, vol. 36, no. 2, pp. 128–39, 2000.
[17] M. Sloman and E. Lupu, "Security and management policy specification," *IEEE Network*, no. March/April, pp. 10–19, 2002.
[18] A. Galis, A. Tan, J. Serrat, E. Salamanca, J. Vivero, Y. Nikolakis, S. Denazis, J. L. Manas, and J. H. Laarhuis, "Policy-based network management for active networks," in *ICT2001*, (Bucarest), 2001.
[19] P. Trimintzios, I. Andrikopoulos, G. Pavlou, C. Cavalcanti, D. A. Goderis, Y. T'Joens, P. Georgatsos, L. Georgiadis, D. Griffin, R. Egan, C. Jacquenet, and G. Memenios, "An architectural framework for providing QoS in IP differentiated services networks," in *Proceedings of the 7th IFIP/IEEE Integrated Management Symposium (IM'01)*, (Seattle, USA), pp. 17–34, 2001.
[20] IETF, "Policy framework," 13 Sept 1999.
[21] T. Hamada, D. C. Blight, and P. Czezowski, "Active policies in knowledge hyperspace: Intelligent agents and policy-based networking," *KNOM review*, vol. 2, no. 2, pp. 31–41, 1999.
[22] IETF, "The COPS (Common Open Policy Service) Protocol," January 2000.
[23] P. Flegkas, P. Trimintzios, G. Pavlou, I. Andrikopoulos, and C. Cavalcanti, "Policy-based extensible hierarchical network management," in *Proceedings of Workshop on Policies for Distributed Systems and Networks (Policy 2001)*, (Bristol, U.K), pp. 230–46, Spinger-Verlang LNCS-1995, Bristol, UK, 2001.
[24] IETF, "Policy core LDAP schema," August 2002.