Identity Analytics - "User Provisioning" Case Study: Using Modelling and Simulation for Policy Decision Support

Marco Casassa Mont, Adrian Baldwin, Simon Shiu

HP Laboratories
HPL-2009-57

Keyword(s):

Identity Analytics, IAM, User Provisioning, Modelling, Simulation, Identity Management, Policy Decision Support

Abstract:

This paper extends and complements paper [24] by providing additional details on how modelling and simulation can support the (policy) decision making process, for Identity and Access Management (IAM). Specifically, the process of making IT (security) policy decisions, within organizations, is complex: it involves reaching consensus between a set of stakeholders (key decision makers, e.g. CISOs/CIOs, domain experts, etc.) who might have different views, opinions and biased perceptions of how policies need to be shaped. This involves multiple negotiations and interactions between stakeholders. IAM is a rich area that introduces various dilemmas, e.g. in terms of required IT investments and related policies. We focus on the "user account provisioning process" for enterprise applications and services, a key IAM feature that has an impact on security, compliance and business outcomes. Whilst security and compliance experts might worry that ineffective policies for provisioning could fuel security and legal threats, business experts might be against policies that dictate overly strong or bureaucratic processes as they could have a negative impact on productivity. Policy decision support tools and methods can firstly help an individual stakeholder to test, refine their understanding of the situation and, secondly, to support the formation of consensus by helping stakeholders to share their assumptions and conclusions. We argue that an approach based on modeling and simulation can help with both these aspects, moreover we show that it is possible to integrate the assumptions made so that they can be directly contrasted and discussed. We explore the associated policy decision making process from these different perspectives and show how our systems modeling approach can provide consistent or comparable data, explanations, "what-if" predictions and analysis at different levels of abstractions. We discuss the implications that this has on the actual IT (security) policy decision making process, for IAM. In this context, we introduce and discuss a fully working Demos2k model for "user account provisioning".

External Posting Date: March 21, 2009 [Fulltext] Approved for External Publication
Internal Posting Date: March 21, 2009 [Fulltext]

Identity Analytics
"User Provisioning" Case Study: Using Modelling and Simulation for Policy Decision Support

Marco Casassa Mont, Adrian Baldwin, Simon Shiu

Hewlett-Packard Labs, Systems Security Lab, Bristol, UK
{marco.casassa-mont, adrian.baldwin, simon.shiu}@hp.com

Abstract. This paper extends and complements paper [24] by providing additional details on how modelling and simulation can support the (policy) decision making process, for Identity and Access Management (IAM). Specifically, the process of making IT (security) policy decisions, within organizations, is complex: it involves reaching consensus between a set of stakeholders (key decision makers, e.g. CISOs/CIOs, domain experts, etc.) who might have different views, opinions and biased perceptions of how policies need to be shaped. This involves multiple negotiations and interactions between stakeholders. IAM is a rich area that introduces various dilemmas, e.g. in terms of required IT investments and related policies. We focus on the "user account provisioning process" for enterprise applications and services, a key IAM feature that has an impact on security, compliance and business outcomes. Whilst security and compliance experts might worry that ineffective policies for provisioning could fuel security and legal threats, business experts might be against policies that dictate overly strong or bureaucratic processes as they could have a negative impact on productivity. Policy decision support tools and methods can firstly help an individual stakeholder to test, refine their understanding of the situation and, secondly, to support the formation of consensus by helping stakeholders to share their assumptions and conclusions. We argue that an approach based on modeling and simulation can help with both these aspects, moreover we show that it is possible to integrate the assumptions made so that they can be directly contrasted and discussed. We explore the associated policy decision making process from these different perspectives and show how our systems modeling approach can provide consistent or comparable data, explanations, "what-if" predictions and analysis at different levels of abstractions. We discuss the implications that this has on the actual IT (security) policy decision making process, for IAM. In this context, we introduce and discuss a fully working Demos2k model for "user account provisioning".

1 Introduction

The process of defining IT (Security) policies within organization is complex. Key decision makers make the final policy decisions, but these are reached through a consensus-building process, involving stakeholders and experts from security, business, financial, legal and HR. It is a considerable challenge to help this diverse group bring their skills and perspectives to the discussion, whilst limiting conflicts and misunderstandings. The main contribution of this paper is to show how modeling and simulation can support the policy decision making process by allowing stakeholders to convey consistent explanations and predictions to different audiences, at the right levels of abstraction.

We illustrate this by means of an IAM case study. IAM is important for protecting and securing the organizations' resources, enabling the right people to access legitimate resources for the right purposes. It is a rich area in terms of the policies that could be defined. In this context, IAM is also a business enabler and has a direct impact on business applications and services. At the very core, IAM solutions [22] provide provisioning, enforcement and auditing capabilities. In short IAM policy decisions have a direct impact in terms of people behaviors, costs, productivity, losses and availability.

We focus on a core IAM capability, the *provisioning process of user accounts* to enterprise applications and services. The provisioning process might be automated or be carried out on ad-hoc basis. It might be subject to failures and/or it might be bypasses, if it is ineffective: depending on its accuracy and reliability, people could get unauthorized (or unnecessary) accesses to resources or be prevented to access legitimate resources. The relevant policies might, for example, dictate levels of automation to be achieved by enterprise provisioning processes, acceptable accuracy levels, required approval and configuration times and number of authorization requests that are necessary, depending on the context and types of resources to be accessed/protected. What are the consequences of setting particular policy decisions? Which people have relevant knowledge or concerns? How do we capture and use their inputs?

The remaining part of this paper is structured as follows: Section 2 expands on our analysis of the policy decision making process, specifically in an IAM context. Section 3 provides further details about enterprise identity management and the provisioning process. Section 4 illustrates how modeling and simulation approaches can effectively help to support the policy decision process. Section 5 describes, in more details, our approach and methodology along with an overview of the specific model we have built for the provisioning process, related simulations and the types of results and analysis that can be provided to the stakeholders.

simulations and the types of results and analysis that can be provided to the stakeholders. Finally, Sections 6, 7 and 8 discuss related work, next steps and conclusions.

This paper extends and complements paper [24] with additional material and details, including a full working Demos2k model [17,18,19], provided in Appendix A. Our work has been carried out in the context of the HP Labs' Identity Analytics activity [11, 26, 27], aiming at providing mechanisms to key decision makers (CIOs/CISOs, domain experts, etc) to support their decision making process.

2 On The Policy Decision Making Process

The motivations for changing or analyzing security policy can come for a number of reasons: a large number of policy "exception" requests are usually a good sign that something is wrong. It can also be any of the stakeholders (i.e. decision makers and domain experts) feeling that the inherent trade-offs are inappropriate, for example IT operations may feel the burden/resources required to maintain a particular policy is too large, or conversely a security officer may feel the threat environment has changed and so a tighter policy is warranted. In these cases either the policy can be changed, or investments and resources can be re-aligned to more efficiently meet the policy. There are numerous challenges to helping the stakeholders, with relevant concerns and subject matter expertise, to express and share their knowledge.

It is important to analyse, in more details, which steps are involved in the process of making decisions about policies. In general, an organization might already have a few (IT) policies in place. Auditing, feedbacks or direct experiences from the fields might periodically indicate that there are policy compliance issues. In this case decisions need to be made on how to improve the situation.

Currently most of these decisions are based on intuitions, common sense, existing literature/cases and inputs coming from experts in the field. In some cases, decisions might only require acting on existing "levers" (such as further investments in IT solutions, education of personnel, monitoring and punishment, etc.) in the hope this will steer the direction towards policy compliance. In other cases decisions might actually require some policy changes (or the definition of new policies). The effectiveness of these changes is then assessed in the future. Similar process happens in case new policies need to be introduced, from scratch. Figure 1 shows the decision making framework that this structured sharing must support, i.e. allowing the stakeholders to reach one of these forms of conclusion.

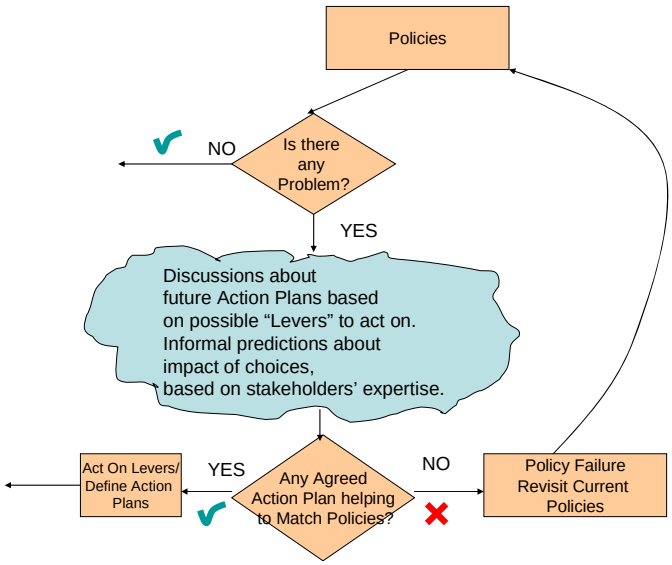

Figure 1. Basic Policy Decision Process

A main theme of this paper is to explore and illustrate how systems modelling [16,17] can provide this kind of support, see Figure 2. We argue that systems modelling can provide more rigor and scientific bases to the process of analysing policies, allowing different stakeholders to understand the current situation and exploring the impact of making policy choices.

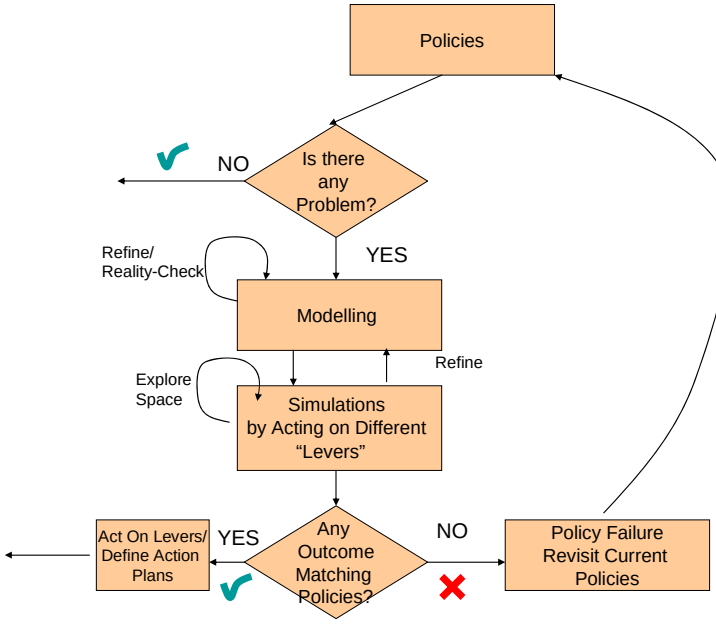

Figure 2. Policy Decision Making Support

We show how a combination of executable process models, probability theory and Monte Carlo style experimentation (based on simulations) can be used to help stakeholders explore their own intuitions and assumptions, share these with others in a coherent and consistent way and jointly investigate the consequences of investments and policy changes. Specifically, models can be used to represent relevant aspects of the reality, including processes involving systems and people. The representation of external and internal events, their likelihood to happen, the initial state, along with cause-effects and related probabilities can be used to explore the impact of choices/decisions. Repeated simulations generate statistically relevant outcomes.

Building models requires iterations, refinements and reality checking with various stakeholders. It can require time and resources. However, once these models are validated and trusted, they can be used not only to explain (complex) aspects affecting the current situation, but also for predictive and "what-if" analysis, by exploring hypothesis and different assumptions. In our vision, these properties suit with the requirement of being more effective and analytic in the process of policy decision making.

Specifically, in the policy decision making process, modelling and simulation techniques can be used to:

- *help stakeholders to assess the current situation*, by means of models representative of the reality, which are potentially coherent with expectations (or could positively challenge them), that reflect current observed measures/metrics and illustrate them by means of different views/perspectives at the right level of abstraction – starting from common and consistent assumptions;
- *help stakeholders to predict the outcomes of acting on different "levers"* (i.e. exploring the space of available choice options), along with the impact these choices have on agreed metrics;
- *bring together different stakeholders' inputs*, in a consistent way, present outputs at the right levels of abstractions and speed the decision making process up.

As a significant example, we are going to explore this in the context of an IAM case study, focusing on provisioning processes. The next section provides some additional background about IAM.

3 Enterprise Identity Management

Identity and Access Management (IAM) solutions for enterprises [22] include functionalities such as authentication, single-sign-on (SSO), authorization, auditing, compliance and assurance management, provisioning, data storage, link to legacy systems and data consolidation. Figure 3 shows the main IAM components and functionalities.

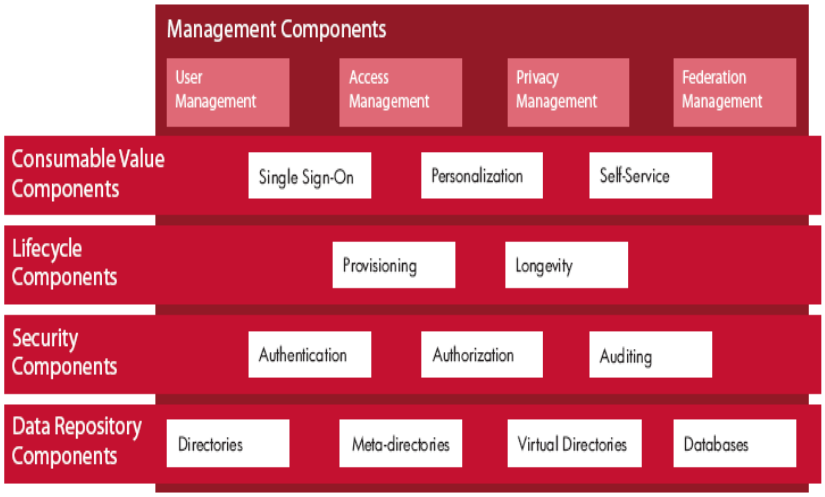

Figure 3. Enterprise Identity Management

Identity management functionalities are, in general, used for user account and access control management, federated identity management and privacy management. A more detailed description of various components and related capabilities is available [22].

For the purpose of this paper, we focus on *user account provisioning solutions*. These solutions are used by enterprises to deal with the lifecycle management of user identities and accounts on protected resources, including the enrolment, customization, modification and removal of user accounts associated with users, employees and customers along with setting rights, permissions and access control information. Getting the right *provisioning* in place is as important as getting the right *enforcement* (authentication, authorization and access control) in place. A wrong or lousy provisioning process could give more than necessary rights to users or prevent them from accessing legitimate resources. This is an IAM area that is still in evolution, along as well as the related processes of defining enterprise users' roles and access control permissions.

At the very core, user account provisioning solutions aims at ensuring that valuable resources (such as business applications and services) are protected against unauthorized accesses. Provisioning processes keep into account changes in the workforce (i.e. people joining, leaving, changing their roles) and organizational changes (re-organisations, large lay-offs, M&As, etc.).

Provisioning of user accounts (and access control permissions) in enterprises usually requires dealing with two core phases:

- ***Approval phase***: the creation, modification or removal of user accounts (associated to a user, for a specific application/service) need to be authorized by one or more people that have managerial responsibilities (e.g. line managers or supervisors);
- ***Deployment and configuration phase***: in case of a successful approval, this phase consists in carrying out configuration activities, to actually create, modify or remove a user account on a system/application/service, along with related user rights.

Depending on the kind of adopted provisioning solution, there might be different degrees of automation, ranging from *ad-hoc, manual processes* to *fully automated and centralized processes*. The former might rely on human interactions and system administrators. The latter might involve the execution of workflows and automated configuration scripts. These phases could have degrees of failures or different implementations, depending on cultural attitudes and working environments. A typical set of *IAM provisioning policies* might be expressed as:

- *P1*: Employees' user accounts should be provisioned within an organization in max 3 days
- *P2*: No user account must be provisioned without management approval
- *P3*: All user accounts to be provisioned (added, modified, changed) on core business applications and services must require 2 levels of approval
- *P4*: Users accounts of people leaving a company must be removed within 2 days the departure date

- *P5*: The accuracy of the provisioning process (in terms of correctly configured user accounts on protected resources) should never be less than 0.99%

The CIO, CISO or maybe a risk manager (decision makers) would be responsible for defining these policies and their appropriateness. However, policy analysis and decisions will require the input and consent ("buy in") of several stakeholders, including:

- **security experts**, that understand the vulnerability of the provisioning process and can articulate the technical consequences**;**
- **business experts and application/service owners**, that understand the criticality of appropriate access to business objectives, and to some extent the business burden the policies create;
- **compliance experts**, that are driven by the need to be compliant to internal guidelines, laws and legislation (such as SOX), being able to pass auditing sessions, etc.;
- **HR experts**, that have an understanding of how the population of employees is evolving over time, which roles they might have and which organisations they work for.

4 Policy Decision Support for Provisioning Management

The policy decision support challenge for *IAM provisioning* is how to allow the different stakeholders to convey their knowledge and concerns. To focus this discussion, we assume a situation where there is some centralized automation provisioning for enterprise applications, but that many applications still maintain "ad-hoc" manual provisioning processes (e.g. carried out by local system administrators). The *security/compliance manager* (domain expert) feels intuitively that more applications should adopt the automated process because she believes it will improve risk and compliance issues. Formally, the *security manager* will be challenged to produce a business case (perhaps a cost-benefit analysis) for the investment, informally there will be a lot of negotiation involving application owners and IT operations (other domain experts). Specifically, the *application owners* will be concerned about disruption to user (aka business) productivity and the IT operations team about the costs and burden that any changes require.

We argue that modeling and simulation can support the overall decision making process. Our aim is to produce a model of the IAM provisioning systems (and related processes) deployed in the organization that will show how to help these stakeholders express and explore their subjective concerns. A useful first step is to identify the different (high-level) *metrics* that these stakeholders will be interested in:

- **Security/Compliance Officer**
  - **Access Accuracy**: the number of correctly configured user accounts, against the overall number of created accounts, including badly configured accounts and hanging accounts;
  - **Approval Accuracy**: the number of approved provisioning activities, against the overall provisioning activities, including the unauthorized ones.

- **Application Owner (Business)**
  - **Productivity Cost**: these are the costs, in terms of loss of productivity (for employees), due to delays during the approval and configuration/deployment phases of the provisioning process.

- **IT Operations (IT Budget Holder)**
  - **IAM Provisioning Cost**: this is the cost of deploying (IAM) automated provisioning solutions, for a specified timeframe (involved license fee, fixed and variable costs);
  - **Provisioning Effort**: this is the actual number of provisioning "transactions" carried out by the organization, in a specific timeframe, giving an idea of the effort and involved workload.

With these metrics in mind we can build an executable process model of the provisioning systems (see Appendix A). A high-level schematic of this model is shown in Figure 4. This high-level model includes representations of:

- **External Events**, including people joining, leaving and changing roles and dependencies on affected applications;
- **Model of IAM Provisioning Processes**, including affected applications, involved approval and configuration/deployment phases and related failures and delays. Input information includes various probability

distributions related to aspects of these processes, as previously described. Measures are collected about the evolution, over time, of these processes and stored in the **State**;

- **Threats** of relevance affecting IAM provisioning processes and/or fuelled by the executions (and failures) of these processes;
- **State**, tracking both low-level measures and derived high-level metrics.

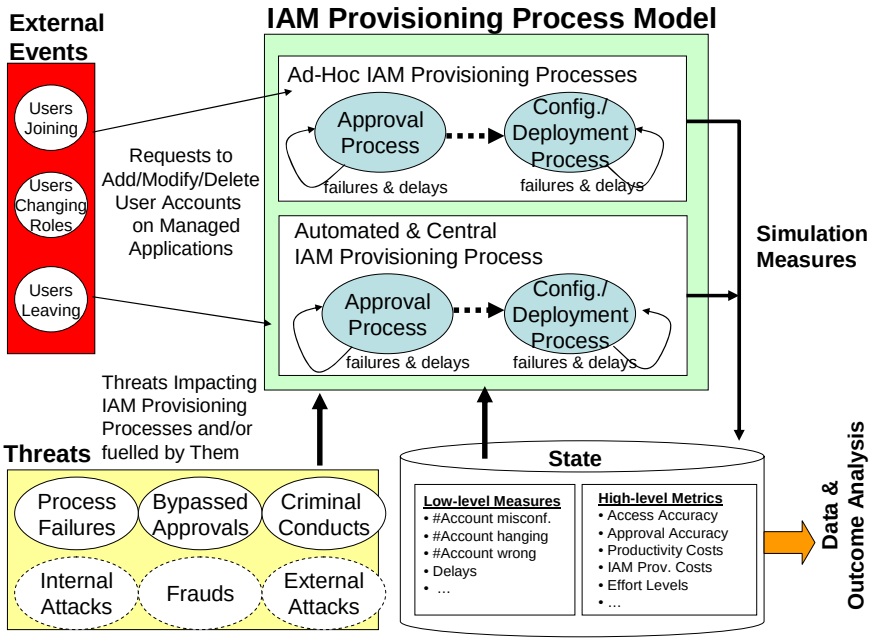

Figure 4. High-level Provisioning Model

More details about the model are provided in Section 5 and in Appendix A, but roughly we (mathematically) model the actual approval and deployment processes. As they execute they affect the model state, which reflect the metrics we are interested in.

These processes are triggered by external events (e.g. employees joining or leaving the organization or changing their role, hence requiring their user accounts to be updated) which we represent stochastically. A simulation, based on the model, proceeds by sampling relevant probability distributions which determine when the external events cause the execution of provisioning processes.

By repeating this simulation many times (i.e. in the style of Monte Carlo analysis), we start to build a picture of how different assumptions (e.g. about how processes execute, how often they are triggered or fail) can affect the measures and metrics we are interested in. The threat processes can be folded into this analysis to explore specific failure or attack situations.

Low-level *measures* (that are used to calculate the high-level *metrics* mentioned above) are tracked by the model and calculated during simulations, including:

- Number of correctly configured and mis-configured user accounts;
- Number of hanging accounts (people that left);
- Overall approval time (delays) for provisioning requests;
- Overall configuration/deployment time (delays);
- Number of lost approval and deployments/configuration requests;
- Number of bypassed approval processes.

For example, Figure 5 shows the probability density functions (pdf functions) of some of these measures, as determined by simulations of our model, over a chosen period of time (e.g. a year):

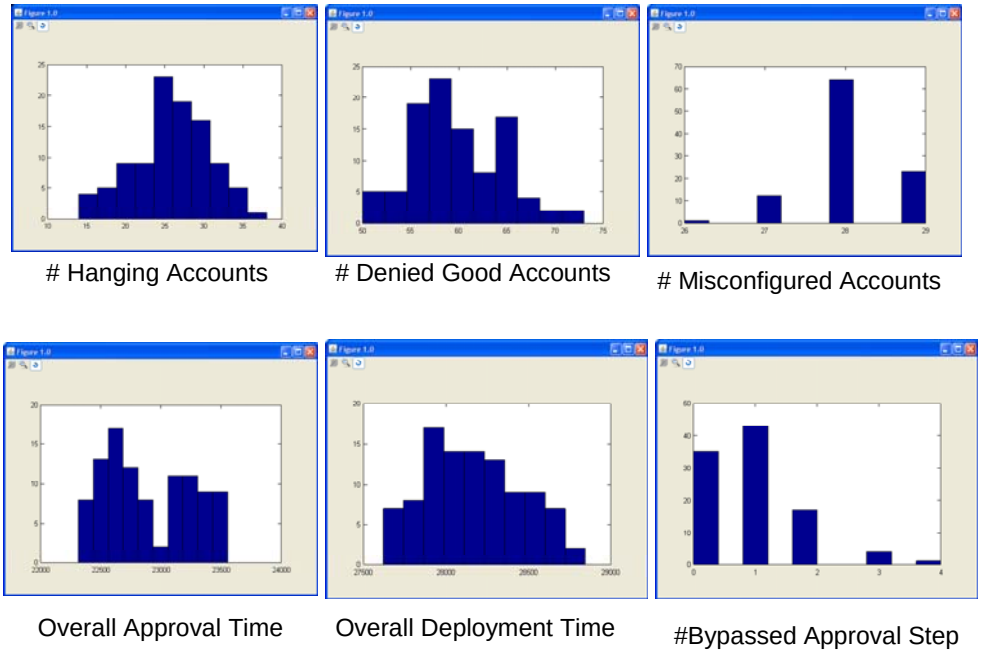

Figure 5. Experimental Results: Pdf of Low-level Measures

The different stakeholders are well placed to compare these fine-grained results with their tacit knowledge, and in some cases with empirical data. A typical next step for an interested stakeholder is to understand and challenge how these results are being derived, e.g. posing the questions "what is it in the assumptions that leads to these results?", and "do I agree with them?".

In addition to supporting this exploration it is important that the model provides a meaningful aggregated view where all the stakeholders can coherently discuss their inputs. The aggregated view should also be meaningful to the key decision maker(s). The graph in Figure 6 illustrates an example of how this may be done, by means of the high-level *metrics*, derived from low-level *measures*.

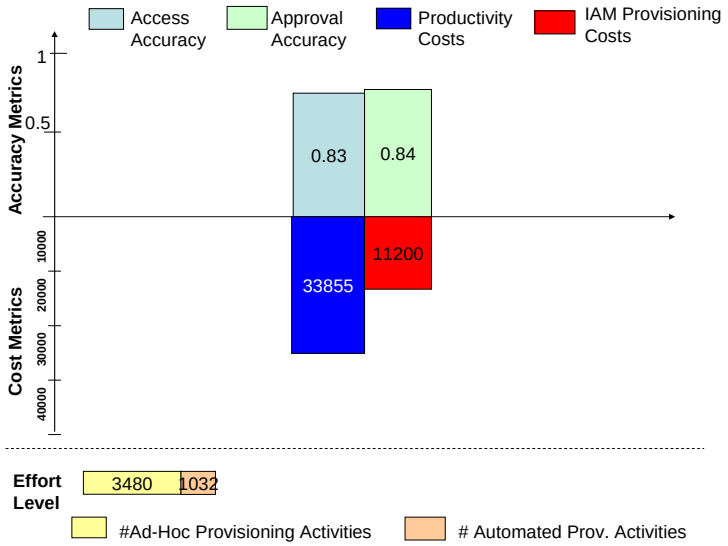

Figure 6. Experimental Results – High-level Metrics

The *cost and accuracy metrics* (shown in Figure 6) may vary depending on the number of provisioning work items, and so the view shows the results for the assumed (modeled) effort level. Section 5 provides additional details for the approach used to produce this normalized view and how to calculate these *metrics*. The key point

though is that the assumptions about how this normalization is done are transparent, and potentially subject to discussions.

In our case study, we consider the case where the enterprise has 5 core business applications and 100 non-core, lower-priority applications. In the *current state*, only 2 core applications and 10 non-core applications are provisioned with automated and centralized IAM processes. Again, Figure 5 and 6 show the *measures* and *metrics* that represent the implications of current enterprise investments in IAM provisioning processes (simulated over a year timeframe).

These figures indicate lack of policy compliance (see policy examples in Section 3). For example, policy *P5* is violated as "access accuracy" is far smaller that 99%.

In an attempt to be compliant, the stakeholders might want to explore the impact of introducing more IAM provisioning automation for protected resources (core and non-core applications/services), by running them under centralized, common processes rather than on an ad-hoc basis. This is one of the *"levers"* a decision maker can act on to change the current situation. Hence, the stakeholders might want to investigate the implications of automating additional applications, in a year timeframe, by considering different automation cases, as shown in Figure 7.

Experiments	**Core Business Applications (5 Apps)**	**Non Core Business Applications (100 Apps)**
CASE #1 – Provisioning CURRENT SITUATION	automation: 2 Apps ad-hoc: 3 Apps	automation: 10 Apps ad-hoc : 90 Apps
CASE #2	automation: 3 Apps ad-hoc : 2 Apps	automation : 40 Apps ad-hoc : 60 Apps
CASE #3	automation: 4 Apps ad-hoc : 1 Apps	automation : 70 Apps ad-hoc : 30 Apps
CASE #4	automation: 5 Apps ad-hoc : 0 Apps	automation: 100 Apps ad-hoc: 0 Apps

Figure 7. Experiments - "What-if" Cases

Simulations of the model can be carried out for each case of interest and the results can be compared. The outcomes, in terms of high-level *metrics*, are shown in Figure 8. This figure shows that accuracy measures are increasing by investing more in automation of IAM provisioning processes. Similarly, productivity costs decrease but IAM provisioning costs increase.

This shows that, for certain values of the "automation lever" (e.g. case #4 - full provisioning automation) the corresponding IAM investment costs are too high, compared to the productivity costs. Further analysis of which applications require more provisioning or different assumptions about future workload might change this analysis.

The point is that these metrics can be used to qualitatively and quantitatively show the impact of policy choices. Similarly, results indicate that Policy *P2* (see Section 3) will never be met (the approval accuracy is always less than 1); hence policy *P*2 might need to be changed.

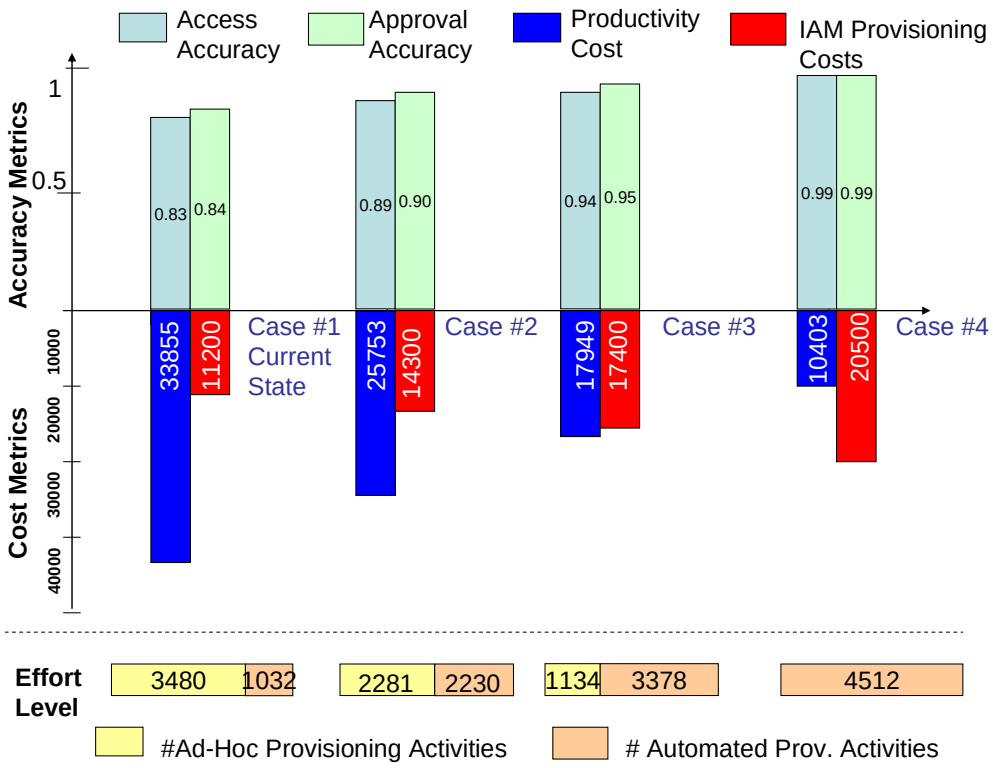

Figure 8. Experiments – Prediction Outcomes for Different "What-if" Cases

5 Our Modeling Approach

Our modeling approach relies on mathematical models and related simulations. The use of mathematical models in engineering has a long and distinguished record of success ranging over mechanical, civil, environmental and electrical/electronic engineering areas. The mathematical methods used in these fields are mainly concerned with continuous phenomena and typically use techniques from calculus such as differential equations. For modeling security and identity management operations the appropriate mathematical methods are more discrete, being drawn from algebra, logic, theoretical computer science and probability theory. In order to apply these methods, we require a conceptual analysis of the relevant aspects of the systems of interest.

The basic methodology that we have adopted, based on the scientific method and tailored to security and identity management, involves hypothesizing a theory or model that explains the current situation. We iterate with this model starting with some observational facts about the current scenario and validating results against experts' opinions and other observed facts. We can gradually add detail to the model based on further observations, e.g. through user interviews, in order to ensure the output of the model sufficiently reflects the current scenario. We can then use the model to explore specific phenomena by varying the assumptions, or adding additional facets representing controls.

In the context of this methodology, we have used a specialized simulation-oriented language, Demos2k [17,18,19], which implements a modelling framework based on the mathematical foundations of a synchronous calculus of resources and processes, together with an associated modal logic. Because of its strong mathematical foundations and sound semantics, we have assurance that simulations based on the models developed in the Demos2k language are robust and reliable - thus, meaningful observations can be taken. The code is executed via repeated experimental simulations in the specially developed experimental environment, where statistically significant information is gathered. The mathematical framework behind the Demos2K programming language revolves around four key concepts:

- **resources,** capturing the essentially static components of the system;
- **processes,** capturing the dynamic components of the system;
- **location,** capturing the spatial distribution and connectivity of the system;
- **environment,** within which a system functions.

In the context of IAM Provisioning processes, "resources" are any valuable asset or element we might want track in the model. For example, this could include core and non-core applications/services, along with related user accounts etc. Modelled "processes" include, among other things, ad-hoc and automated provisioning processes, inclusive of the approval and deployment/configuration phases. "Location" modelling aspects are also of particular importance: they represent spatial distribution aspects of applications, local (regional) people attitudes and habits and localized instantiation of IAM processes. Finally, the "environment" aspect is used to model additional characteristics of the scenario under observation that are of relevance for the simulation steps, e.g. existing threats.

Specifically, in the IAM provisioning case study, we model the difference between *ad-hoc* and *centralized IAM provisioning* and explore the impact of choices on existing policies and/or to shape new policies. We seek to illustrate this through the impact on the *measures* and *metrics*, introduced in Section 4.

Figures 9 and 10 provide additional details about our model, which keeps into account *external events* (user joining, leaving and changing roles) and *enterprise IAM provisioning processes*. Specifically each type of provisioning activity - triggered by an external event, involves (a) a user and one or more applications/services and (b) is explicitly modeled by means of a modeling "process" (Figure 9).

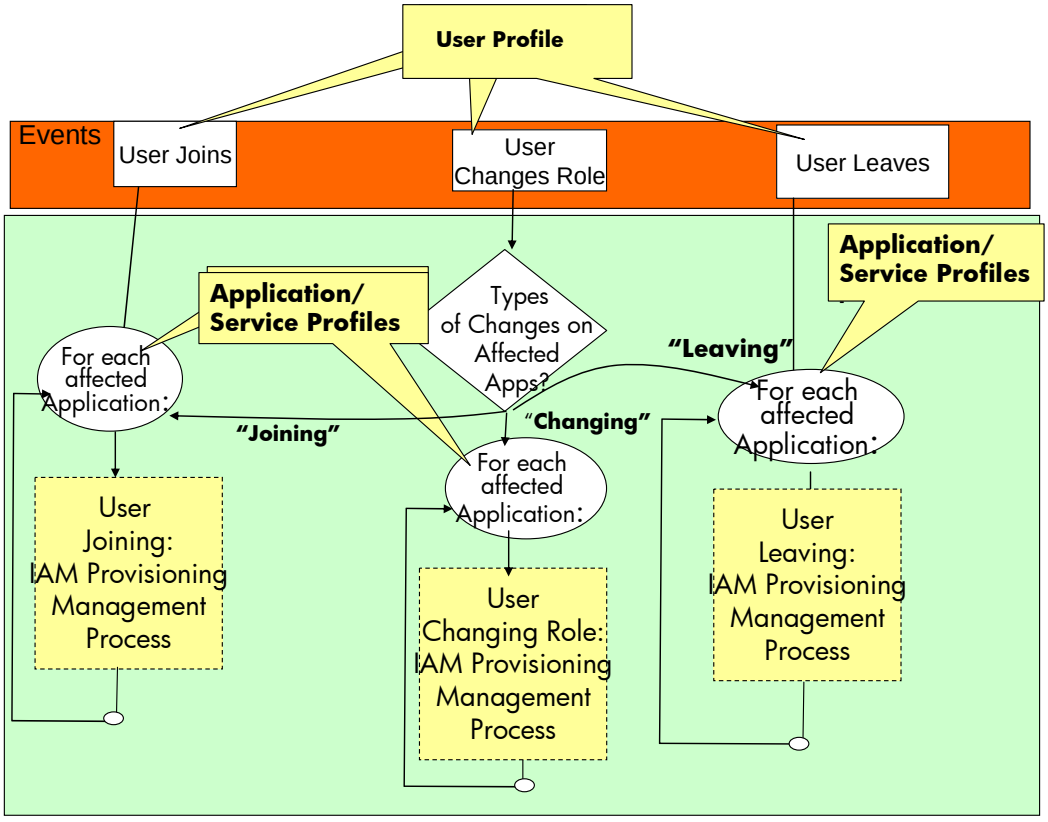

Figure 9. Discrete-event Probabilistic Model – IAM Provisioning Processes

Figure 10 provides the details of the modeled "provisioning workflow" for "*users joining*" the organization: this includes approval and deployment phases, delays and failures (including bypassing the system) along with the points where measurements are taken. The impact of the approval and configuration/deployment phases are explicitly considered. This includes keeping track of:

- The likelihood of failures of approval requests (e.g. managers in charge of approving requests do not actually do it, or the request is lost) and deployment activities (due to probabilistic faults or lack of activity of system administrators);
- The fact that the approval process can, under some circumstances, be bypassed.

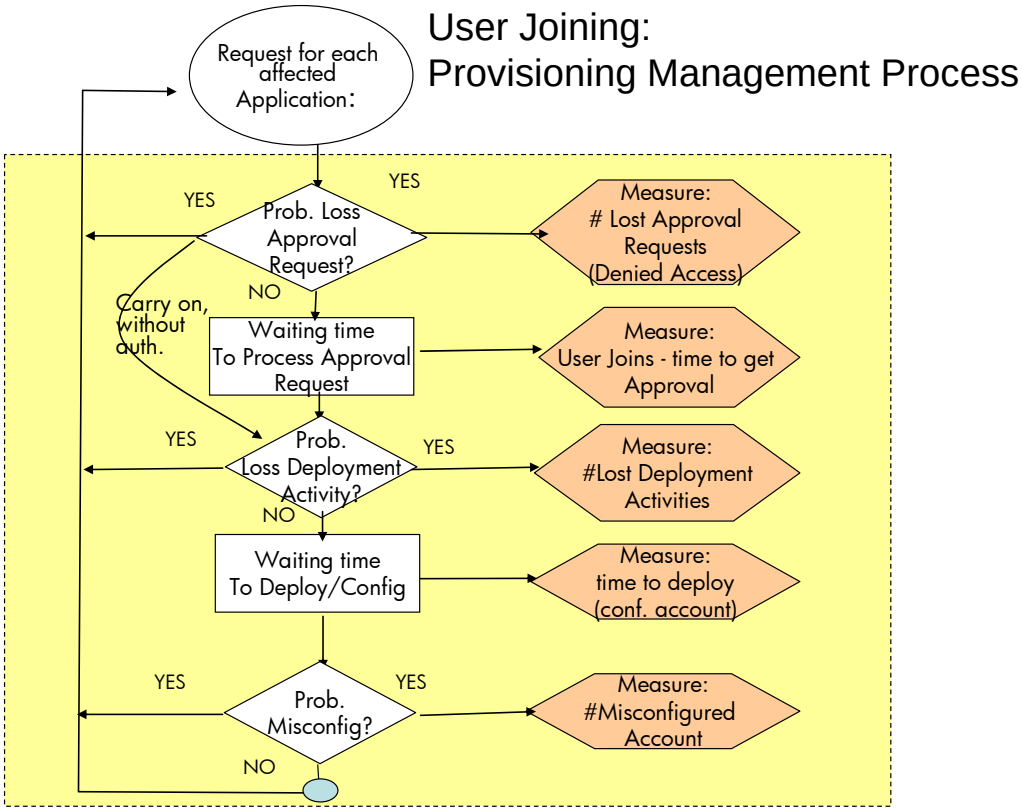

Figure 10. Schematic of the Executable Process Model for IAM Provisioning - New Users Joining an Organisation

The outcomes of this "provisioning workflow" are stored by means of model variables (representing low-level measures), including:

- Time delays;
- Number of misconfigured user accounts;
- Number of denied user accounts (that users are entitled to);
- Number of bypassed approval processes.

Similar comments apply for our modeling of the provisioning processes involved for "*user changing roles*", Figure 11 and "*users leaving*", Figure 12.

In case of user changing roles, in addition to updating the affected user accounts (e.g. due to changes of permissions and rights), some related provisioning activities might also involve *creating new user accounts on applications* (e.g. for users that were previously not entitled to use them) or *removing user accounts from applications* (e.g. for users that are not anymore entitled to use them).

For users leaving an organisation, it is unlikely that users might try to bypass the approval process, in case of failures. Hence user accounts might be left "hanging", with all the negative consequences (from security and compliance perspective) that this can bring. This is reflected in the modelled process, shown in Figure 12.

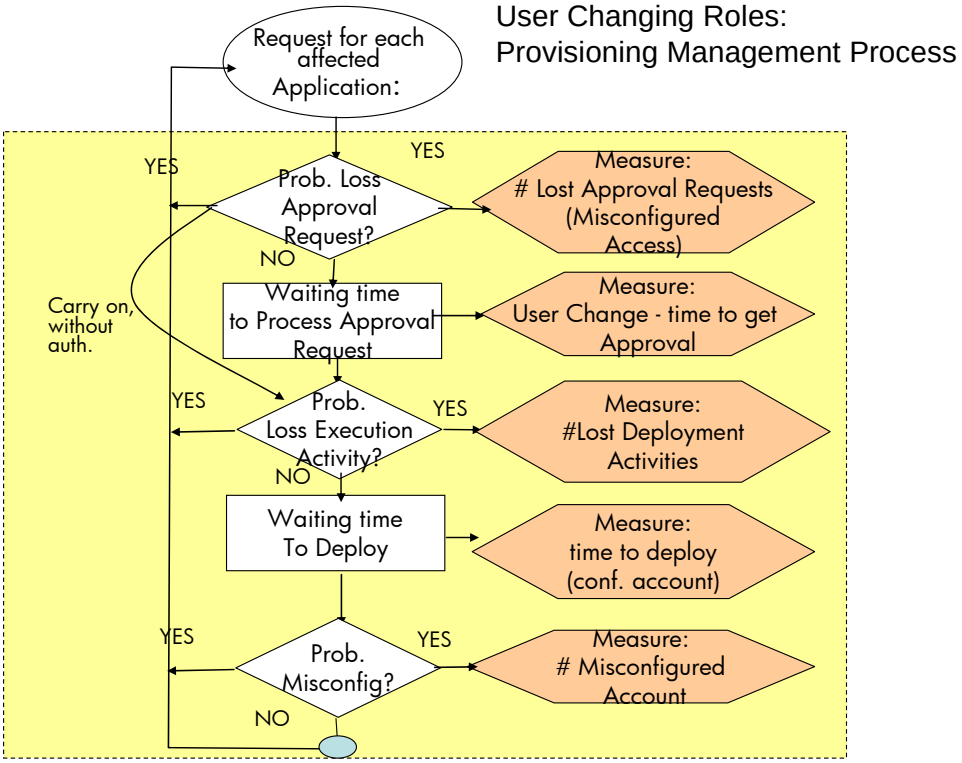

Figure 11. Schematic of the Executable Process Model for IAM Provisioning – Users Changing Role in an Organisation

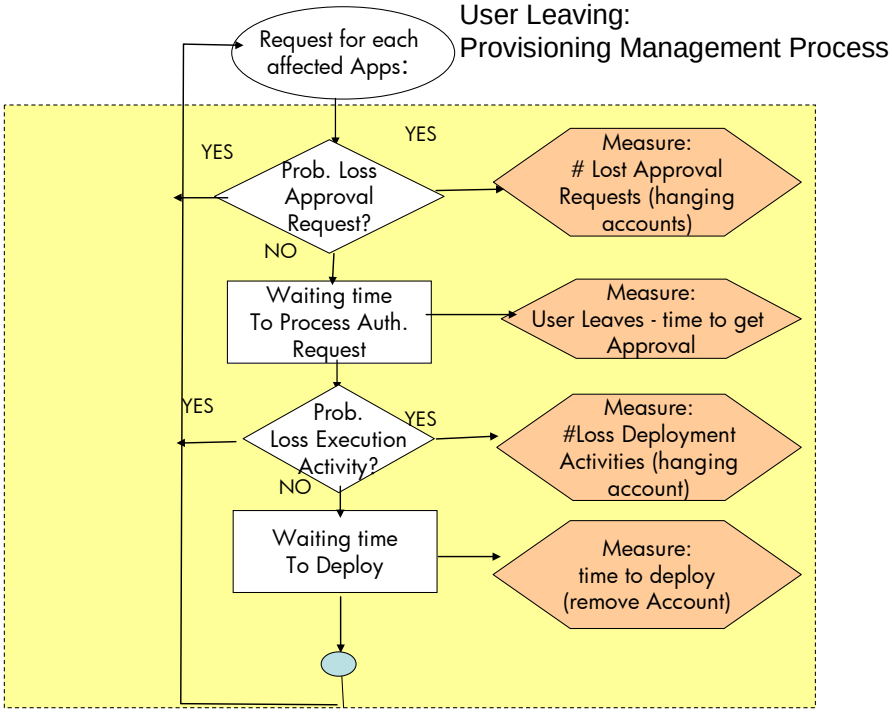

Figure 12. Schematic of the Executable Process Model for IAM Provisioning – Users Leaving an Organisation

More details about the implemented Demos2k model follow.

External events, such as the arrival of a new user, are modeled stochastically, i.e. with appropriate probability distributions. Figure 13 provides the list of the core External Events handled by our model (also see Appendix A).

External Event	Definition (Probability Distribution)	Description
numUserJoinPerPeriodTime	negexp(6)	Number of users joining an organisation per period of time (1 week). Modeled as a negative exponential, with means 6.
numUserLeavePerPeriodTime	negexp(3)	Number of users leaving an organisation, per period of time (1 week). Modeled as a negative exponential, with means 3.
numUserChangePerPeriodTime	negexp(4)	Number of users changing role in an organisation per period of time (1 week). Modeled as a negative exponential, with means 4.

Figure 13. Model – Details about the Definition of External Events

Intuitively, the more IAM provisioning processes are centralized, automated and managed under common policies the more their behaviours are similar, as opposed to ad-hoc processes. However, the more centralization and automation is introduced, the higher the impact of IAM costs (license fees) and faults. We test and explore this trade-off using a Monte Carlo style simulation which can be run with parameterized assumptions about which applications have automated or ad-hoc provisioning. This allows us to build a picture of how different choices will lead to different outcomes.

An instance of a simulation specifies the number of core and non-core applications and the number of applications having automated and ad-hoc provisioning. Figure 7 shows various assumptions we made in terms of applications and automation levels. Within the model, there is a range of parameters determining the probability distributions for how often the different processes are triggered (typically varying means on negative exponentials), and probability distributions for which applications are affected by the different user centric processes.

Additional key information *received in input by the model* includes:

- Number of core and non-core applications;
- For each of the above two types of applications, the number of applications that are managed with ad-hoc provisioning processes and the number of applications that are managed with IAM automated provisioning;
- Probability distributions indicating the number of core and non-core applications affected by users joining, leaving or changing roles. These are point uniform distributions (pud) that depend on the role of the users;
- Average profiles of *ad-hoc* and *IAM automated* provisioning processes. For each of these two categories of processes, the following parameters are provided:
    - Waiting time for Approval Request: modeled as a normal distributions;
    - Probability of Loss of Approval Request: modeled as a Bernoulli test;
    - Probability of Bypassing the Approval Process: calculated dynamically as a Bernoulli test where the probability of the event is: *1 – 1/(1 + num approval failures)*. The more "approval failure" happens, the higher is the probability this test succeeds. This might be particularly true in case of centralised IAM provisioning processes.
    - Deployment/Configuration time (to actually provision a user account on an application): modeled as a normal distribution;
    - Probability of Loss of Deployment/Configuration Phase: modeled as a Bernoulli test;
    - Probability of User Account misconfiguration: modeled as a Bernoulli test.

The actual definition of these *parameters*, provided in *input to the model*, is shown in Figure 14:

Input Parameter	Definition (Probability Distribution)	Description
numApps[TIER1_APP]	Number (e.g. 5)	Overall number of Tier 1 (core) enterprise applications/services.
numApps[TIER2_APP]	Number (e.g. 100)	Overall number of Tier 2 (non-core, secondary) enterprise applications/services.
numAppsWithCentralIAMProvisioning[TIER1_APP]	Number (e.g. 2)	Number of Tier 1 enterprise applications/services subject to central, automated IAM provisioning.

numAppsWithCentralIAMProvisioning[TIER2_APP]	Number (e.g. 10)	Number of Tier 2 enterprise applications/services subject to central, automated IAM provisioning.
p_AppWithCentralIAMProvisioning[TIER1_APP]	binom (1, numAppsWithCentralIAMProvisio-ing[TIER1_APP]/numApps[TIER1_APP])	Probability that a Tier1 application/service has a central IAM provisioning. Modelled as a Bernoulli test.
p_AppWithCentralIAMProvisioning[TIER2_APP]	binom (1, numAppsWithCentralIAMProvision-ing[TIER2_APP]/numApps[TIER2_APP]	Probability that a Tier2 application/service has a central IAM provisioning. Modelled as a Bernoulli test.
p_userRole	pud[(0.7,CLERK_ADMIN),(0.1,MANAGER),(0.01,EXECUTIVE), (0.05,ITSTAFF), (0.1, SALES_MARKETING), (0.01, HR), (0.03, RESEARCH_DEVEL)];	Probability for a user (joining, leaving, changing role) of having a specific role within the organisation. Modeled by means of a point uniform probability distribution.

p_numTier1ReqApps[<ROLE>]	p_numTier1ReqApps[CLERK_ADMIN] = pud[(0.65, 1), (0.3, 2), (0.05,3)];
	
	p_numTier1ReqApps[MANAGER] = pud[(0.65, 2), (0.3, 3), (0.05,4)];
	
	p_numTier1ReqApps[EXECUTIVE] = pud[(0.65, 2), (0.3, 3), (0.05,4)];
	
	p_numTier1ReqApps[ITSTAFF] = pud[(0.65, 2), (0.3, 3), (0.05,2)];
	
	p_numTier1ReqApps[SALES_MARKETING] = pud[(0.65, 3), (0.3, 4), (0.05,5)];
	
	p_numTier1ReqApps[HR] = pud[(0.65, 1), (0.3, 2), (0.05,3)];
	
	p_numTier1ReqApps[RESEARCH_DEVEL] = pud[(0.65,3), (0.3, 4), (0.05,5)];
		Probability for a user (joining, leaving, changing role) of having to use a specific number of Tier 1 applications/services, given their role. Modeled by means of point uniform probability distributions.

p_numTier2ReqApps[<ROLE>]	p_numTier2ReqApps[CLERK_ADMIN] = pud[(0.65, 4), (0.3, 5), (0.05,6)];
	
	p_numTier2ReqApps[MANAGER] = pud[(0.65, 5), (0.3, 6), (0.05,7)];
	
	p_numTier2ReqApps[EXECUTIVE] = pud[(0.65, 5), (0.3, 6), (0.05,7)];
	
	p_numTier2ReqApps[ITSTAFF] = pud[(0.65, 5), (0.3, 6), (0.05,7)];
	
	p_numTier2ReqApps[SALES_MARKETING] = pud[(0.65, 6), (0.3, 7), (0.05,8)];
	
	p_numTier2ReqApps[HR] = pud[(0.65, 4), (0.3, 5), (0.05,6)];
	
	p_numTier2ReqApps[RESEARCH_DEVEL] = pud[(0.65, 7), (0.3, 8), (0.05,9)];
		Probability for a user (joining, leaving, changing role) of having to use a specific number of Tier 2 applications/services, given their role. Modeled by means of point uniform probability distributions.

p_UserChange_ProvisActivity_PerApplication	pud[(0.1, USER_JOIN), (0.8, USER_CHANGE), (0.1,USER_LEAVE)]	In case of a user changing role, probability that an affected application needs to be provisioned with: changes of the existing user account; or a user account needs to be removed; or a user account needs to be added. Modeled by means of point uniform probability distributions.
waitingTimeMgmtAp-	normal(2,1)	IAM-enabled (central and

proval_IAM_AutomatedProvisioning		automated) provisioning process. Probability of waiting (days) during the provisioning approval phase. Modeled with a normal probability distribution.
probLossApprovalRequest_IAM_AutomatedProvisioning	binom (1, 1/500)	IAM-enabled (central and automated) provisioning process. Testing the loss of an approval request. Modeled with a Bernoulli test.
probBypassApprovalProcess_IAM_AutomatedProvisioning	binom (1, 1 - (1/(NumLossIAMProvisioningApprovalRequest+1)))	IAM-enabled (central and automated) provisioning. Testing if the approval process is bypassed. Modeled with a Bernoulli test. The involved probability is determined at runtime, based on the approval process failure rate.
probLossExecutionActivity_IAM_AutomatedProvisioning	binom (1, 1/500)	IAM-enabled (central and automated) provisioning process. Probability that a provisioning deployment request (execution phase) is loss. Modeled with a Bernoulli test.
ConfigDeploymentTime_IAM_AutomatedProvisioning	normal(1,1)	IAM-enabled (central and automated) provisioning process. Probability of waiting (days) during the provisioning deployment/ configuration phase. Modeled with a normal probability distribution.
probMisconfiguration_IAM_AutomatedProvisioning	binom (1, 1/500)	IAM-enabled (central and automated) provisioning process. Probability that the provisioning deployment/ configuration phase created misconfigured accounts. Modeled with a Bernoulli test.
waitingTimeMgmtApproval_AdHoc_Provisioning	normal(5,3)	Ad-Hoc (local) provisioning process. Probability of waiting (days) during the provisioning approval phase. Modeled with a normal probability distribution.
probLossApprovalRequest_AdHoc_Provisioning	binom (1, 1/4)	Ad-Hoc (local) provisioning process. Testing the loss of an approval request. Modeled with a Bernoulli test.
probBypassApprovalProcess_AdHoc_Provisioning	binom (1, 1 - (1/(NumLossAHProvisioningApprovalRequest+1)))	Ad-Hoc (local) provisioning process. Testing if the approval process is bypassed. Modeled with a Bernoulli test. The involved probability is determined at runtime, based on the approval process failure rate.
probLossExecutionActivity_AdHoc_Provisioning	binom (1, 1/10)	Ad-Hoc (local) provisioning process. Probability that a provisioning deployment request (execution phase) is loss. Modeled with a Bernoulli test.
ConfigDeploymentTime_AdHoc_Provisioning	normal(7,3)	Ad-Hoc (local) provisioning process. Probability of waiting (days) during the provisioning deployment/ configuration phase. Modeled with a normal probability distribution.
probMisconfiguration_AdHoc_Provisioning	binom (1, 1/10)	Ad-Hoc (local) provisioning process. Probability that the provisioning deployment/ configuration phase created misconfigured accounts. Modeled with a Bernoulli test.

Figure 14. Model – Details about the Definition of Input Parameters

The complete definition of probability distributions (means, variances, etc.), for all input parameters, is provided in Appendix A, along with a copy of our model.

It is important to notice that some of the probability distributions mentioned above have been tuned, within our model, based on empirical values provided by customers and HP business groups. They can be modified to reflect the reality of specific provisioning processes. The current model has been kept simple: it can be further refined and extended, depending on the level of details needed or available.

As anticipated in Section 4, the model can keeps track of cumulative *measures* and provide them in output. Figure 15 provides a list of these output variables (also see Appendix A):

Output Variables - Measures	**Description**
NumApprovalRequest	Overall number of approval requests
NumLossIAMProvisioningApprovalRequest	Number of lost approval requests for central IAM provisioning processes
NumLossAHProvisioningApprovalRequest	Number of lost approval requests for Ad-Hoc provisioning processes
NumLossApprovalRequest	Overall number of lost approval requests (sum of the above two measures)
CarryOnDespiteNoApproval	Overall number of bypassed approval processes
OverallTimeApproval	Overall approval time
SuccessNumApprovalRequest	Overall number of successfully processes approval requests
NumLossDeployment	Overall number of lost deployment/configuration activities
OverallTimeDeployment	Overall deployment/configuration time
SuccessNumDeployment	Overall number of successful deployment activities
NumMisconfigAccess	Overall number of misconfigured user accounts
NumDeniedGoodAccess	Overall number of denied, legitimate user accounts (due to legitimate user account that have not been created/enabled)
NumWrongAccess	Overall number of wrong user accounts, that should not exist (due to hanging user accounts)
OngoingProvisActivities	Overall number of ongoing provisioning activities, over time (i.e. not yet fully completed)

Figure 15. Model – Details about Output Variables - Measures

The model also provides *detailed measures* for each type of provisioning activity (i.e. user joining, leaving or changing roles), comprehensive of the impact of centralised and ad-hoc provisioning activities – see Appendix A.

The above measures ("low-level measures" in Section 4) keep track of the impact of managing provisioning processes for all types of managed events. The model uses these measures to derive the *high-level metrics* introduced in Section 4. A complete list of these measures, along with their definition is provided in Figure 16 (also see Appendix A).

Metrics	Formula	Description
Access Accuracy	1-(w1*UAD+w2*UAM+w3*UAH)/ (UAA)	w1, w2, w3 are relevance weights in the [0,1] range, UAD is the number of denied user accounts, UAM is the number of misconfigured user accounts, UAH is the number of hanging user accounts and UAA is the overall number of user account provisioned (for which either there has been approval or the approval process has been bypassed);
Approval Accuracy	#Approved_Provisioning / (#Approved_Provisioning + # Bypassed_Approvals)	
Productivity Costs	[(join_appr_time+ change_appr_time) + (join_prov_time + change_prov_time)] * *Unit_cost_per_day* + [(#loss_join_appr + #loss_join_prov) + (#loss_change_appr+#loss_change_prov)] **Unit_cost_lost.*	keeps into account loss of productivity due to waiting time (for the approval and deployment phases) and for lost of approval and deployment activities. The impact of these costs are weighted by constants for "unit cost per day" and "unit cost per loss".
IAM Automation Cost	Fixed_Costs + Variable_Costs*Num_IAM_Automated_Apps	Estimated *costs of running* automated IAM provisioning processes, depending of fixed costs (e.g. fixed yearly *fee) and variable* costs (e.g. additional license fees depending on the number of provisioned applications)
IAM Effort	#IAM_automated_provisioning_activities	
Ad-hoc Effort	#Ad-Hoc_provisoning_activities	

Figure 16. Modelling - Definition of Metrics

Experiments have been carried out by running simulations (by executing 100 times the same model), over a predefined period of time (e.g. 1 year). These simulations produce, as an outcome, statistically significant low-level *measures* and related high-level *metrics*. This information can be processed, analysed and eventually displayed, as shown in Figures 5, 6, 8. Experiments can be reconfigured in a straightforward way, by changing the simulated time frame and/or the number of times a model needs to be executed.

This model can be run by different stakeholders (decision makers and domain experts) to directly carry out "what-if" experiments, by acting on available "levers" and changing model parameters.

Stakeholders can focus on low-level measures or high-level metrics, depending on the desired level of abstraction they work at, compare results across multiple "what-if experiments" and, if required, dig down the details (e.g. up to the level of the probability density functions of output measures/metrics).

This enables stakeholders to improve their understanding of the overall aspects involved in a specific scenario, map predicted outcomes to current policies and compare against their intuitions; it provides them with additional evidence to back their opinions and positions.

6 Related Work

The concept of using scientific input in policy decision making has been explored in various papers, in specific areas such as hydrology, land usage and environmental contexts [1,2,3] or social science [4]. This work, however, does not illustrate how this can be achieved in practice by using modeling and simulation, specifically to address the needs of different stakeholders operating at different levels of abstraction.

The area of policy decision support for security, privacy and identity management has not yet been widely explored. A case for using modeling and simulation in information security is made in [5]. Paper [23] explores risk metrics for identity management but it uses a traditional bottom-up risk management approach, based on the assessment of auditing metrics.

Modelling and simulation have been used in specific contexts of identity management and privacy, to explore the impact of technical choices on policies, such as password policies [6,7], identity phishing [8] and security polices for network access control [9]. This is important related work. However, it does not describe how to effectively provide support to different stakeholders in the policy decision making process and focuses just on a few aspects of identity management.

Our work aims at exploring and advancing the state of the art in this space, for a wide range of IAM aspects. This R&D work is part of the HP Labs Security and Identity Analytics project [10,11]. We are not aware of current research or commercial solutions that aim at modelling and simulating the overall complexity of identity management and related policy decision making process.

Standards such as ISO 27001 [12], CoBit [13], ITIL [14] describe best practices and methodologies respectively in terms of information security management, IT governance and service management. Decision makers still need to understand, interpret and instantiate them in their specific operational environments. We can use these standards as drivers and references but our work adds the value of grounding the reasoning to specific environments, related policies and the underlying IT infrastructures (possibly along with human and social behaviours).

Our work is complementary to studies on policy refinement and deployment. These studies (e.g. [25]) primarily focus on how to refine policies, once they have been agreed, in order to enforce them. We focus on the policy decision making process and how to support it.

We leverage the work done by HP Labs in the Open Analytics project [15,16], that we consider as a reference. Specifically, we use Demos2k [17,18,19] as the reference tool for our modelling and simulation activities. Finally, an important aspect of our work is the studies in the space of economics and social science. We aim to leverage work done in [20] to build mathematical models that realistically reflect users' behaviours and the associated impact.

7 Discussion and Future Work

We have implemented a fully working model of an IAM provisioning management process along with measures, metrics and analysis of outcomes of relevance to different stakeholders. It has been (internally) tested to support the policy decision making process in the IAM provisioning space. This model can be extended in various directions. More detailed descriptions of IAM provisioning processes can be introduced (if information is available) along with a representation of user behaviours (e.g. [11]), to explore, for example, their impact during the approval and deployment phases, on regional and cultural basis. The enforcement side of IAM (e.g. authentication, authorization, etc.) can also be factored in to explore investments trade-offs, based on (policy) choices and various assumptions made by stakeholders. Initial work in this space is described in [21]. Further areas to be investigated include the modeling of the impact of security threats on IAM processes (and in particular for provisioning processes), involved risks and how to support related policy decision making processes.

Our future R&D work includes exploring additional IAM areas (where support could be provided for policy decision making), including: enterprise single-sign-on, authorization and authentication, auditing, IAM outsourcing, IAM-as-a-Service and implications of IAM in cloud computing and Web 2.0 scenarios.

Ultimately, the goal is to create a model library, covering key, relevant IT aspects and policy concerns in the IAM area that can be systematically leveraged by decision makers and domain experts. To achieve this, we are looking for opportunities to engage with HP customers (and other parties) in technology trials, to further validate our approach (to support the policy decision making process) against their current approaches, refine our models and methodology.

8 Conclusions

This paper describes current challenges in making effective policy decision within organisations, both in terms of how to form good opinions and then dealing with painful politics and the process of reaching consensus. We illustrated how modeling and simulation methods help to address these aspects, providing objective and relevant analysis for all the involved stakeholders at appropriate levels of abstractions. We focused an IAM provisioning scenario, where relevant (and conflicting) policies might apply. We illustrated how the outcomes of our modeling and simulation activities, based on "what-if" analysis, can explain and predict the impact of specific (policy) choices, from different viewpoints.

This is work in progress. We will engage in customer trials to further tune our approach and models. Part of this work will be carried out in the context of the HP Labs' Identity and Security Analytics project [10,11].

9. References

1. Becu, N., Neef, A., Schreinemachers, P., Sankapitux, C.: Participatory computer simulation to support collective decision making: Potential and limits of stakeholder involvement, ScienceDirect, Elsevier, 2007
2. Adams, P.W., Hairston, A.B.: PUsing Scientific Input in Policy and Decision Making, Oregon State University, 1995
3. Khoo, H.H., Spedding, T.A., Tobin, L., Taplin, D.: Integrated Simulation and Modelling Approach to Decision Making and Environmental Protection, Kluwer Academic Publisher, 2001
4. Kennedy, C., Theodoropoulos, G.: Towards Intelligent Data-Driven Simulation for Policy Decision Support in the Social Sciences, School of Computer Science, University of Birmingham, UK, 2005
5. Saunders, J.H.: The Case for Modeling and Simulation of Information Security, GSEC National Defense University, http://www.johnsaunders.com/papers/securitysimulation.htm, 2001
6. Shay, R., Bhargav-Spantzel, A., Bertino, B.: password policy simulation and analysis, DIM 2007, 2007
7. Adams, A, Sasse, M.A.: Users are not the enemies, Communications of the ACM, 1999
8. Moore, T., Clayton, R.: The Consequence of Non-Cooperation in the Fight Against Phishing, 3rd APWG eCrime Researchers Summit, 2008
9. Koh, J.Y, Yi, M., Cho, T, Kim, H., and Kim. H.: Knowledge-Based Modeling and Simulation of Network Access Control Mechanisms Representing Security Policies, Springer, Information and Communications Security LNCS book, 2002
10. Security Analytics: HP Labs, SSL, http://www.hpl.hp.com/research/systems_security.html, 2008
11. Casassa Mont, M., Baldwin, A., Shiu, S.: On Identity Analytics: Setting the Context, HPL TR, HPL-2008-84, 2008
12. ISO: ISO 27001, Information Security Management, , 2005
13. ISACA: Cobit, IT Governance, http://www.isaca.org/, 2008
14. ITIL: ITIL IT Infrastructure Library for Service Management, http://www.itil-officialsite.com/home/home.asp, 2008
15. Pym, D., Taylor, R., Tofts, C., Yearworth, M., Monahan, B., Gittler, F.: Systems and services sciences: a rationale and a research agenda (Open Analytics Project), HPL-2006-112, 2006
16. Taylor, R., Tofts, C.: Model Based Services Discovery and Management, PICMET 2008, 2008
17. Demos2k: Demos 2k, http://www.demos2k.org/, 2000
18. Birtwistle, G.: Demos, discrete event modelling on Simula. Macmillian, 1979
19. Monahan, B.: DXM - The Demos eXperiments Manager, HP Labs Technical Report, 2008
20. Trust Economics: UK DTI grant P0007, Trust Economics Project, 2008
21. Baldwin, A., Casassa Mont, M., Monahan, B., Pym, D., Shiu, S.: System Modelling to Support Economic Analysis of Security Investments: A case Study in Identity and Access Management, submitted to conference, 2009
22. Casassa Mont, M., Bramhall, P., Pato, J.: On Adaptive Identity Management: The Next Generation of Identity Management Technologies - HPL-2003-149, 2003
23. Peterson, G.: Introduction to Identity Management Risk Metrics, IEEE Security & Privacy, 2006
24. Casassa Mont, M., Baldwin, A., Shiu, S.: Using Modelling and Simulation for Policy Decision Support in Identity Management, submitted to IEEE Policy 2009, HPL Technical Report, HPL-2009-56, 2009
25. Sloman, M., Dulay, N., Nuseibeh, B.: SecPol: Specification and Analysis of Security Policy for Distributed Systems, 1997
26. Casassa Mont, M., Baldwin, A., Griffin, J., Shiu, S., Beres, Y.: Identity Analytics: Using Modeling and Simulation to Improve Data Security Decision Making- HPL-2008-188, 2008
27. Casassa Mont, M., Baldwin, A., Griffin, J., Shiu S.,: Towards Identity Analytics in Enterprises - HPL-2008-186, 2008

Appendix A: “User Account Provisioning” Model

This appendix provides the entire code of a fully working “User Account Provisioning” model, developed with Demos2k [17,18,19] .
This model has been used to carry out various simulations and experiments, as described in Sections 4 and 5.

```
(*  Author: Marco Casassa Mont
   Date: 02 March 2009
   Version: 08

   Model of IAM provisioning:

   - Users: can join a company, leave, change role. To perform their jobs need to access a set of applications/services
   - Applications/Services: enterprise assets enabling business functions
            - different types of apps/services based on their value/importance
            - different access control management approach: centralised/ad-hoc
   - Users roles: they determine what users can do in an organisations and what they can access
            - Are defined within an organisation
            - Could be leveraged for automating provisioning management (provisioning, configuration, etc.)
   - Users accounts: created on systems hosting apps/services. They identify a user on a system (id credentials).
              They are associated to users' rights/permissions, based on users' roles
              - can be manually managed by sys admins
              - can be automatically managed with provisioning management solutions
   - Sys admins: in charge on handling systems hosting applications.
         Might be asked to create users' accounts and give them access rights
   - Managers: in charge of managing users. Might authorise/deny users' access to apps/services based on their roles
   - Provisioning Management solutions: automate the process of dealing with the automated management of user accounts and rights
            - provisioning/deprovisioning of users' accounts on systems hosting apps/services
            - configuration of user accounts with rights/permissions based on their roles
            - handling the workflow process for Approvals & deployment/change management

   Assumptions
   1) A set S1 of apps/services are centrally managed within the organisation:
     - Central HR repository defines users' roles, apps/services they are allowed to access and their rights/permissions
     - Automated provisioning management solutions handle:
       - Approval workflow, in case users join/leave/change roles. Managers are actively involved. Sys admins are not.
       - configuration management (creation/removal of user accounts) & provisioning of access rights

   2) A set S2 of apps/services are managed on ad-hoc basis within the organisation:
     - Local decisions/rules based on users' roles define which apps/services users are allowed to access and their rights/permissions
     - Managers still need to authorise but could be bypassed by interacting with sys admins
     - Ad-hoc provisioning management implemented by sys admins:
       - manual Approval workflow (emails, interactions with managers), in case users join/leave/change roles
       - manual configuration management (creation/removal of user accounts) & provisioning of access rights

   Low-level Measures (related to provisioning activities for user joining, leaving and changing roles)
     - number of user accounts correctly configured;
     - number of mis-configured user accounts;
```

```
      - number of hanging accounts;
      - overall approval time (delays) for provisioning requests;
      - average (company-wide) approval time per provisioning request;
      - overall configuration/deployment time (delays);
      - average (company-wide) deployment and configuration time, per provisioning request;
      - number of lost approvals and deployments/configuration;
      - number of bypassed approval processes

   High-Level metrics
    - access accuracy
    - approval accuracy
    - productivity costs
    - IAM automation costs
    - IAM automation effort
    - Ad-Hoc effort

 *)

//* LIVELOCK-STEPS : 10000
//* SPAWN-LIMIT : 10000

// Timescaling constants
//---------------------------
cons days       = 1;       // time unit = days
cons hrs        = days/24;
cons hours      = days/24;  // alternative spelling
cons mins       = hrs/60;
cons secs       = mins/60;
cons msecs      = secs/1000;

cons weeks      = 7 * days;
cons months     = 4 * weeks;
cons years      = 365 * days;
cons centuries  = 100 * years;


//simulation time constants
cons simulationAdvancementTimePeriod = days;
cons observedTimePeriod = months;
cons simulationTimeframe = years;

//dxm parameters
cons noparam = 0;

//General constants
//------------------

// Application Types

cons TIER1_APP = 0;  // Critical business apps
cons TIER2_APP = 1;  // Secondary-level business apps
```

```
cons USER_JOIN = 0;
cons USER_LEAVE = 1;
cons USER_CHANGE = 2;

// Events
//---------------
cons numUserJoinPerPeriodTime = negexp(6);
cons numUserLeavePerPeriodTime = negexp(3);
cons numUserChangePerPeriodTime = negexp(4);

// Business Locations/Regions
//---------------------------------------
cons EMEA = 0;  // Europe, Middle East, Africa
cons AM = 1;    // Americas
cons APJ = 2;   // Asia, Pacific and Japan


// User Profile
//---------------

//Probability distribution defining users' roles

cons CLERK_ADMIN = 0;
cons MANAGER = 1;
cons EXECUTIVE  = 2;
cons ITSTAFF  = 3;
cons SALES_MARKETING = 4;
cons HR = 5;
cons RESEARCH_DEVEL  = 6;

cons p_userRole= pud[(0.7,CLERK_ADMIN),(0.1,MANAGER),(0.01,EXECUTIVE), (0.05,ITSTAFF), (0.1, SALES_MARKETING), (0.01, HR), (0.03, RESEARCH_DEVEL)];

//Probability distribution defining number of apps that users need to access, based on (1) their Roles (2) Application Tiers

cons p_numTier1ReqApps[CLERK_ADMIN] = pud[(0.65, 1), (0.3, 2), (0.05,3)];
cons p_numTier1ReqApps[MANAGER] = pud[(0.65, 2), (0.3, 3), (0.05,4)];
cons p_numTier1ReqApps[EXECUTIVE] = pud[(0.65, 2), (0.3, 3), (0.05,4)];
cons p_numTier1ReqApps[ITSTAFF] = pud[(0.65, 2), (0.3, 3), (0.05,2)];
cons p_numTier1ReqApps[SALES_MARKETING] = pud[(0.65, 3), (0.3, 4), (0.05,5)];
cons p_numTier1ReqApps[HR] = pud[(0.65, 1), (0.3, 2), (0.05,3)];
cons p_numTier1ReqApps[RESEARCH_DEVEL] = pud[(0.65,3), (0.3, 4), (0.05,5)];

cons p_numTier2ReqApps[CLERK_ADMIN] = pud[(0.65, 4), (0.3, 5), (0.05,6)];
cons p_numTier2ReqApps[MANAGER] = pud[(0.65, 5), (0.3, 6), (0.05,7)];
cons p_numTier2ReqApps[EXECUTIVE] = pud[(0.65, 5), (0.3, 6), (0.05,7)];
cons p_numTier2ReqApps[ITSTAFF] = pud[(0.65, 5), (0.3, 6), (0.05,7)];
cons p_numTier2ReqApps[SALES_MARKETING] = pud[(0.65, 6), (0.3, 7), (0.05,8)];
cons p_numTier2ReqApps[HR] = pud[(0.65, 4), (0.3, 5), (0.05,6)];
cons p_numTier2ReqApps[RESEARCH_DEVEL] = pud[(0.65, 7), (0.3, 8), (0.05,9)];


cons p_userLocation = pud[(0.5,EMEA),(0.4,AM),(0.1,APJ)];
```

```
// Application profiles
//-------------------------

cons numApps[TIER1_APP] = 5;  // set of Tier1 applications that employees might need to access, depending on their roles
cons numApps[TIER2_APP] = 100; // set of Tier2 applications that employees might need to access, depending on their roles

cons numAppsWithCentralIAMProvisioning[TIER1_APP] = 2; //Tier1 applications with central IAM Provisioning solutions
cons numAppsWithCentralIAMProvisioning[TIER2_APP] = 10; //Tier2 applications with central IAM Provisioning solutions

cons p_AppWithCentralIAMProvisioning[TIER1_APP] = binom (1, numAppsWithCentralIAMProvisioning[TIER1_APP]/numApps[TIER1_APP]);
cons p_AppWithCentralIAMProvisioning[TIER2_APP] = binom (1, numAppsWithCentralIAMProvisioning[TIER2_APP]/numApps[TIER2_APP]);

cons numOverallApps = numApps[TIER1_APP]  + numApps[TIER2_APP] ; // overall number of apps


//Application Location/Administration

cons p_appManagementLocation = pud[(0.4,EMEA),(0.4,AM),(0.2,APJ)];


// Provisioning Processes
//-----------------------------

//IAM-enabled (central & automated) provisioning process

cons waitingTimeMgmtApproval_IAM_AutomatedProvisioning = normal(2,1);  //days
cons probLossApprovalRequest_IAM_AutomatedProvisioning = binom (1, 1/500);
// cons probBypassApprovalProcess_IAM_AutomatedProvisioning = binom (1, 1 -(1/(NumLossIAMProvisioningApprovalRequest+1)));  - this is dynamically calculated
cons probLossExecutionActivity_IAM_AutomatedProvisioning = binom (1, 1/500);
cons ConfigDeploymentTime_IAM_AutomatedProvisioning = normal(1,1);  //days
cons probMisconfiguration_IAM_AutomatedProvisioning = binom (1, 1/500);


//AD-HOC (AH) provisioning process
cons waitingTimeMgmtApproval_AdHoc_Provisioning = normal(5,3);  //days
cons probLossApprovalRequest_AdHoc_Provisioning = binom (1, 1/4);
// cons probBypassApprovalProcess_AdHoc_Provisioning = binom (1, 1 -(1/(NumLossAHProvisioningApprovalRequest+1)));  - this is dynamically calculated
cons probLossExecutionActivity_AdHoc_Provisioning = binom (1, 1/10);
cons ConfigDeploymentTime_AdHoc_Provisioning = normal(7,3);  //days
cons probMisconfiguration_AdHoc_Provisioning = binom (1, 1/10);

cons p_UserChange_ProvisActivity_PerApplication = pud[(0.1, USER_JOIN), (0.8, USER_CHANGE), (0.1,USER_LEAVE)];

cons uniDist = uniform (0,1);



// Metrics - Constants
//-------------------------------------
cons access_accuracy_UAD_weight = 1; //weight for User Accounts Denied to users
```

```
cons access_accuracy_UAM_weight = 1; //weight for User Accounts Misconfigured
cons access_accuracy_UAH_weight = 1; //weight for User Accounts Hanging

cons unit_cost_per_day = 1;
cons unit_cost_failure = 5;

cons IAM_provisioning_fixed_cost = 10000;
cons IAM_provisioning_variable_cost = 100;

// run control
var demos_sample_tick   = 0;
var done = 0;


// Variables
//--------------

// LOW-LEVEL MEASURES - VARIABLES
//------------------------------------------------------------

var joinNum = 0;
var joinNumApp = 0;
var joinNumApprovalRequest = 0;
var joinNumLossApprovalRequest =0;
var joinCarryOnDespiteNoApproval =0;
var joinOverallTimeApproval =0;
var joinSuccessNumApprovalRequest =0;
var joinNumLossDeployment =0;
var joinOverallTimeDeployment =0;
var joinSuccessNumDeployment =0;

var joinNumMisconfigAccess =0;
var joinNumDeniedGoodAccess =0;

var leaveNum = 0;
var leaveNumApp =0;
var leaveNumApprovalRequest = 0;
var leaveNumLossApprovalRequest =0;
var leaveOverallTimeApproval =0;
var leaveSuccessNumApprovalRequest =0;
var leaveNumLossDeployment =0;
var leaveOverallTimeDeployment =0;
var leaveSuccessNumDeployment =0;

var leaveNumWrongAccess =0;

var changeNum = 0;
var changeNumApp = 0;
var changeNumApprovalRequest = 0;
var changeNumLossApprovalRequest =0;
var changeCarryOnDespiteNoApproval =0;
var changeOverallTimeApproval =0;
```

```
var changeSuccessNumApprovalRequest =0;
var changeNumLossDeployment =0;
var changeOverallTimeDeployment =0;
var changeSuccessNumDeployment =0;

var changeNumMisconfigAccess =0;
var changeNumDeniedGoodAccess =0;

var NumApprovalRequest = 0;          //ASSUMPTION:  a user account is involved in every approval request
                                     //                    this gives an indication of the overall number of accounts involved by Provisioning
var NumLossIAMProvisioningApprovalRequest =0;
var NumLossAHProvisioningApprovalRequest = 0;
var NumLossApprovalRequest =0;
var CarryOnDespiteNoApproval =0;
var OverallTimeApproval =0;
var SuccessNumApprovalRequest =0;
var NumLossDeployment =0;
var OverallTimeDeployment =0;
var SuccessNumDeployment =0;

var NumMisconfigAccess =0;
var NumDeniedGoodAccess =0;
var NumWrongAccess =0;

var OngoingProvisActivities = 0;

// HIGH-LEVEL METRICS - VARIABLES
//-------------------------------------------------------

var Access_accuracy = 0;
var Approval_accuracy = 0;

var Productivity_cost = 0;

var IAM_automation_cost = 0;

var IAM_automation_effort= 0;
var AH_effort = 0;









// CLASSES (DEMOS2k PROCESSES)
// -------------------------------------------------

// Class initialising other classes
```

```
class initialise =
{
   entity (eventJoinGenerator, eventJoinGenerator, 0);
   entity (eventLeaveGenerator, eventLeaveGenerator, 0);
   entity (eventChangeGenerator, eventChangeGenerator, 0);
   entity (measurement, measurement, 0);
   hold(simulationTimeframe);
   done :=1;
}


//**********************
// USER JOINING
//**********************


// class dealing with the generation of relevant events (User Joining, Leaving, Changing role ...)

class eventJoinGenerator =
{
  local var userNum = -1;
   repeat {

      // number of user joining per period of time
      userNum := rnd(numUserJoinPerPeriodTime);
      //trace ("numUserJoinPerDay=%v", userNum);

      do userNum
        {
          //trace ("EVENT USER JOIN");
          entity(userJoinProcess, userJoinProcess, 0);
        }

      hold(weeks);
   }
}

// Process related to managing a new user joining the company
class userJoinProcess =
{
  local var userRole = p_userRole;
  local var numTier1RequiredApps = p_numTier1ReqApps[userRole];
  local var numTier2RequiredApps = p_numTier2ReqApps[userRole];
  local var userLocation = p_userLocation;
  local var appRegion = -1;
  local var IAMProvisioningEnabled = -1;

  joinNum := joinNum +1;

  // managing process to get access to tier1 applications
  do numTier1RequiredApps
   {
```

```
    joinNumApp := joinNumApp+1;
    IAMProvisioningEnabled :=  p_AppWithCentralIAMProvisioning[TIER1_APP];
    appRegion := p_appManagementLocation;

    // counting effort, in terms of IAM automated and adhoc provisioning activities
    try [IAMProvisioningEnabled == 1] then
     {
      IAM_automation_effort:= IAM_automation_effort+1;
     }
    etry[] then
     {
      AH_effort := AH_effort +1;
     }

    entity(userJoinProvisioningManagement, userJoinProvisioningManagement(TIER1_APP,#IAMProvisioningEnabled, #appRegion, #userRole, #userLocation), 0);
  }

   // managing process to get access to tier2 applications
 do numTier2RequiredApps
  {
    joinNumApp := joinNumApp+1;
    IAMProvisioningEnabled :=  p_AppWithCentralIAMProvisioning[TIER2_APP];
    appRegion := p_appManagementLocation;

    // counting effort, in terms of IAM automated and ad-hoc provisioning activities
    try [IAMProvisioningEnabled == 1] then
     {
      IAM_automation_effort:= IAM_automation_effort+1;
     }
    etry[] then
     {
      AH_effort := AH_effort +1;
     }

    entity(userJoinProvisioningManagement, userJoinProvisioningManagement(TIER2_APP,#IAMProvisioningEnabled, #appRegion, #userRole, #userLocation), 0);

  }

}

// User Joining - Provisioning Management per Application
class userJoinProvisioningManagement (aType, emEnabl, aReg, uRole, uLocation) =
{
  local var appType = aType;
  local var userRole = uRole;
  local var userLocation = uLocation;

  local var IAMProvisioningEnabled = emEnabl;
  local var appRegion = aReg;

  local var ApprovalReqStart =0;
  local var ApprovalReqEnd = 0;
```

```
local var provisStart =0;
local var provisEnd = 0;
local var carryOn =0;     // 0: do not carry on
                          // 1: carry on

// dynamical calculus of approvals bypassed, depending on number of previous faults
local var probBypassApprovalProcess_IAM_AutomatedProvisioning = 0;
local var probBypassApprovalProcess_AdHoc_Provisioning = 0;


// trace ("USER JOINING - APP TYPE=%v IAM Provisioning Enabled=%v  ", appType,IAMProvisioningEnabled );

// Starting the provisioning process

joinNumApprovalRequest := joinNumApprovalRequest +1;

//global counters
NumApprovalRequest := NumApprovalRequest +1;
OngoingProvisActivities := OngoingProvisActivities +1;


try [IAMProvisioningEnabled == 1] then
  { // case where the application is centrally managed, with the same predefined processes

    try [probLossApprovalRequest_IAM_AutomatedProvisioning ==1] then
     {

       // loss Approval request. Nothing might happen
       // However ... Chance of bypassing the system (probBypassApprovalProcess_IAM_AutomatedProvisioning)

       probBypassApprovalProcess_IAM_AutomatedProvisioning := uniDist;

       try [ probBypassApprovalProcess_IAM_AutomatedProvisioning < 1 -(1/(NumLossIAMProvisioningApprovalRequest+1)) ] then

         {
           // This is Automated Provisioning Management. The user might nevertheless bypass the Approval process and directly
           // ask the sys admin to get their access rights to the application ...

           // There is an approval loss but the process is bypassed

           carryOn :=1;

           joinCarryOnDespiteNoApproval  := joinCarryOnDespiteNoApproval + 1;

           // there is a loss in Approval, but the process carries on

           joinNumLossApprovalRequest := joinNumLossApprovalRequest +1;

           NumLossApprovalRequest := NumLossApprovalRequest +1;
           NumLossIAMProvisioningApprovalRequest := NumLossIAMProvisioningApprovalRequest +1;
```

```
      // overall counters
      CarryOnDespiteNoApproval  := CarryOnDespiteNoApproval + 1;

    }
  etry[] then
    {

      // There is an approval failure and the approval process has not been bypassed

      joinNumLossApprovalRequest := joinNumLossApprovalRequest +1;
      joinNumDeniedGoodAccess := joinNumDeniedGoodAccess +1;

      // overall counters
      NumLossApprovalRequest := NumLossApprovalRequest +1;
      NumLossIAMProvisioningApprovalRequest := NumLossIAMProvisioningApprovalRequest +1;
      NumDeniedGoodAccess := NumDeniedGoodAccess +1;

    }

 }

etry [] then
 {
   carryOn := 1;

  // Approval happens.
  ApprovalReqStart := DEMOS_TIME;
  hold(waitingTimeMgmtApproval_IAM_AutomatedProvisioning);
  ApprovalReqEnd := DEMOS_TIME;
  joinOverallTimeApproval := joinOverallTimeApproval + (ApprovalReqEnd - ApprovalReqStart);
  joinSuccessNumApprovalRequest := joinSuccessNumApprovalRequest +1;

  // global counters
  OverallTimeApproval := OverallTimeApproval + (ApprovalReqEnd - ApprovalReqStart);
  SuccessNumApprovalRequest := SuccessNumApprovalRequest +1;

 }


try [carryOn ==1] then
 {

  // Approval happened or process has been bypassed

  // Proceeding with provisioning (deployment) phase

  try [probLossExecutionActivity_IAM_AutomatedProvisioning ==1] then
   {
    // probability of loss of configuration deployment/actual provisioning phase
    joinNumLossDeployment := joinNumLossDeployment +1;
    joinNumDeniedGoodAccess := joinNumDeniedGoodAccess +1;
```

```
        // global counters
        NumLossDeployment := NumLossDeployment +1;
        NumDeniedGoodAccess := NumDeniedGoodAccess +1;

      }
     etry [] then
      {

        // provisioning phase/deployment happens
        provisStart := DEMOS_TIME;
        hold(ConfigDeploymentTime_IAM_AutomatedProvisioning);
        provisEnd := DEMOS_TIME;
        joinOverallTimeDeployment := joinOverallTimeDeployment + (provisEnd - provisStart);
        joinSuccessNumDeployment := joinSuccessNumDeployment +1;

        // overall counters
        OverallTimeDeployment := OverallTimeDeployment + (provisEnd - provisStart);
        SuccessNumDeployment := SuccessNumDeployment +1;


        // Has the user account been misconfigured? (incorrect access control settings)
        try [probMisconfiguration_IAM_AutomatedProvisioning ==1] then
        {
          // user account has been misconfigured
          joinNumMisconfigAccess := joinNumMisconfigAccess +1;

          // overall counters
          NumMisconfigAccess := NumMisconfigAccess +1;

        }
        etry[] then
        {
          // user account properly configured
        }
     }
   }

  etry [] then
   {
     // approval process failed and/or not bypassed - no carry on
   }


 }
etry[] then
 { // case where the application is managed in ad-hoc way, with processes that might vary

   try [probLossApprovalRequest_AdHoc_Provisioning ==1] then
    {
      // loss Approval request. Nothing might happen
      // However ... Chance of bypassing the system (probBypassApprovalProcess_AdHoc_Provisioning)
```

```
probBypassApprovalProcess_AdHoc_Provisioning := uniDist;


try [probBypassApprovalProcess_AdHoc_Provisioning < 1 -(1/(NumLossAHProvisioningApprovalRequest+1))] then
  {
    // This is AD-HOC Management. The user might bypass the Approval process and directly
    // ask the sys admin to get their access rights to the application ...

    // There is an approval loss by the process is bypassed

    carryOn :=1;

    joinCarryOnDespiteNoApproval  := joinCarryOnDespiteNoApproval + 1;

    // there is a loss in Approval, but the process carries on

    joinNumLossApprovalRequest := joinNumLossApprovalRequest +1;

    NumLossApprovalRequest := NumLossApprovalRequest +1;
    NumLossAHProvisioningApprovalRequest := NumLossAHProvisioningApprovalRequest +1;

    // overall counters
    CarryOnDespiteNoApproval  := CarryOnDespiteNoApproval + 1;

  }
 etry[] then
  {

     // There is an approval loss and the approval process has not been bypassed

     joinNumLossApprovalRequest := joinNumLossApprovalRequest +1;
     joinNumDeniedGoodAccess := joinNumDeniedGoodAccess +1;

     // overall counters
     NumLossApprovalRequest := NumLossApprovalRequest +1;
     NumLossAHProvisioningApprovalRequest := NumLossAHProvisioningApprovalRequest +1;
     NumDeniedGoodAccess := NumDeniedGoodAccess +1;

  }

}
etry[] then
 {
  carryOn := 1;

 // Approval has been granted
 ApprovalReqStart := DEMOS_TIME;
 hold(waitingTimeMgmtApproval_AdHoc_Provisioning);
 ApprovalReqEnd := DEMOS_TIME;

 joinOverallTimeApproval := joinOverallTimeApproval + (ApprovalReqEnd - ApprovalReqStart);
 joinSuccessNumApprovalRequest := joinSuccessNumApprovalRequest +1;
```

```
  // overall counters
  OverallTimeApproval := OverallTimeApproval + (ApprovalReqEnd - ApprovalReqStart);
  SuccessNumApprovalRequest := SuccessNumApprovalRequest +1;

 }


try [carryOn ==1] then
 {

  // Approval has been granted or process is bypassed

  // Proceeding with provisioning (deployment) phase

  try [probLossExecutionActivity_AdHoc_Provisioning ==1] then
   {
    // probability of loss of configuration deployment/actual provisioning phase
    joinNumLossDeployment := joinNumLossDeployment +1;
    joinNumDeniedGoodAccess := joinNumDeniedGoodAccess +1;

    // overall counters
    NumLossDeployment := NumLossDeployment +1;
    NumDeniedGoodAccess := NumDeniedGoodAccess +1;

   }
  etry [] then
   {

     // provisioning phase/deployment happens
     provisStart := DEMOS_TIME;
     hold(ConfigDeploymentTime_AdHoc_Provisioning);
     provisEnd := DEMOS_TIME;
     joinOverallTimeDeployment := joinOverallTimeDeployment + (provisEnd - provisStart);
     joinSuccessNumDeployment := joinSuccessNumDeployment +1;

     // overall counters
     OverallTimeDeployment := OverallTimeDeployment + (provisEnd - provisStart);
     SuccessNumDeployment := SuccessNumDeployment +1;


     // Has the user account been misconfigured? (incorrect access control settings)
     try [probMisconfiguration_AdHoc_Provisioning ==1] then
     {
       // user account has been misconfigured
       joinNumMisconfigAccess := joinNumMisconfigAccess +1;

       // overall counters
       NumMisconfigAccess := NumMisconfigAccess +1;

     }
     etry[] then
```

```
            {
              // user account properly configured
            }
          }
        }
      etry [] then
       {

       }

     }

  // global counters
  OngoingProvisActivities := OngoingProvisActivities -1;


}


//**********************
// USER LEAVING
//**********************


class eventLeaveGenerator =
{
  local var userNum = -1;
    repeat {

        // number of user leaving per period of Time
        userNum := rnd(numUserLeavePerPeriodTime);
        //trace ("numUserLeavePerDay=%v", userNum);

        do userNum
          {
            //trace ("EVENT USER LEAVING");
            entity(userLeaveProcess, userLeaveProcess, 0);
          }

        hold(weeks);
    }
}


// Process related to managing a new user leaving the company
class userLeaveProcess =
{
  local var userRole = p_userRole;
  local var numTier1RequiredApps = p_numTier1ReqApps[userRole];
  local var numTier2RequiredApps = p_numTier2ReqApps[userRole];
  local var userLocation = p_userLocation;
  local var appRegion = -1;
```

```
  local var IAMProvisioningEnabled = -1;


  leaveNum := leaveNum +1;

  // managing process to get access to tier1 applications
  do numTier1RequiredApps
   {
     leaveNumApp := leaveNumApp+1;
     IAMProvisioningEnabled :=  p_AppWithCentralIAMProvisioning[TIER1_APP];
     appRegion := p_appManagementLocation;

     // counting effort, in terms of IAM automated and adhoc provisioning activities
     try [IAMProvisioningEnabled == 1] then
      {
        IAM_automation_effort:= IAM_automation_effort+1;
      }
     etry[] then
      {
        AH_effort := AH_effort +1;
      }

     entity(userLeaveProvisioningManagement, userLeaveProvisioningManagement(TIER1_APP,#IAMProvisioningEnabled, #appRegion, #userRole,#userLocation), 0);
   }

    // managing process to get access to tier2 applications
  do numTier2RequiredApps
   {
     leaveNumApp := leaveNumApp+1;
     IAMProvisioningEnabled :=  p_AppWithCentralIAMProvisioning[TIER1_APP];
     appRegion := p_appManagementLocation;

     // counting effort, in terms of IAM automated and ad-hoc provisioning activities
     try [IAMProvisioningEnabled == 1] then
      {
        IAM_automation_effort:= IAM_automation_effort+1;
      }
     etry[] then
      {
        AH_effort := AH_effort +1;
      }

     entity(userLeaveProvisioningManagement, userLeaveProvisioningManagement(TIER2_APP,#IAMProvisioningEnabled, #appRegion, #userRole,#userLocation), 0);
   }

}


// User Leaving - Management of Provisioning per Application
class userLeaveProvisioningManagement (aType, emEnabl, aReg, uRole, uLocation) =
{
  local var appType = aType;
```

```
local var userRole = uRole;
local var userLocation = uLocation;

local var IAMProvisioningEnabled = emEnabl;
local var appRegion = aReg;

local var ApprovalReqStart =0;
local var ApprovalReqEnd = 0;

local var provisStart =0;
local var provisEnd = 0;


// trace ("USER LEAVING - APP TYPE=%v IAM Provisioning Enabled=%v  ", appType,IAMProvisioningEnabled );

// Starting the provisioning process

leaveNumApprovalRequest := leaveNumApprovalRequest +1;

// general counter
NumApprovalRequest := NumApprovalRequest +1;
OngoingProvisActivities := OngoingProvisActivities +1;


try [IAMProvisioningEnabled == 1] then
  { // case where the application is centrally managed, with the same predefined processes

    try [probLossApprovalRequest_IAM_AutomatedProvisioning ==1] then
     {
       // loss Approval request. Nothing happens
       // This might really get unnoticed, as the user has levt/got different role ...

       leaveNumLossApprovalRequest := leaveNumLossApprovalRequest +1;
       leaveNumWrongAccess := leaveNumWrongAccess +1;

       // general counter
       NumLossApprovalRequest := NumLossApprovalRequest +1;
       NumLossIAMProvisioningApprovalRequest := NumLossIAMProvisioningApprovalRequest +1;
       NumWrongAccess := NumWrongAccess +1;


     }
    etry [] then
     {
       // Approval happens.
       ApprovalReqStart := DEMOS_TIME;
       hold(waitingTimeMgmtApproval_IAM_AutomatedProvisioning);
       ApprovalReqEnd := DEMOS_TIME;
       leaveOverallTimeApproval := leaveOverallTimeApproval + (ApprovalReqEnd - ApprovalReqStart);
       leaveSuccessNumApprovalRequest := leaveSuccessNumApprovalRequest +1;

       // general counter
```

```
    OverallTimeApproval := OverallTimeApproval + (ApprovalReqEnd - ApprovalReqStart);
    SuccessNumApprovalRequest := SuccessNumApprovalRequest +1;


    // Proceeding with provisioning (deployment) phase

    try [probLossExecutionActivity_IAM_AutomatedProvisioning ==1] then
     {
       // probability of loss of configuration deployment/actual provisioning phase
       leaveNumLossDeployment := leaveNumLossDeployment +1;
       leaveNumWrongAccess := leaveNumWrongAccess +1;

       // general counter
       NumLossDeployment := NumLossDeployment +1;
       NumWrongAccess := NumWrongAccess +1;


     }
    etry [] then
     {

        // provisioning phase/deployment happens
        provisStart := DEMOS_TIME;
        hold(ConfigDeploymentTime_IAM_AutomatedProvisioning);
        provisEnd := DEMOS_TIME;
        leaveOverallTimeDeployment := leaveOverallTimeDeployment + (provisEnd - provisStart);
        leaveSuccessNumDeployment := leaveSuccessNumDeployment +1;

        // general counter
        OverallTimeDeployment := OverallTimeDeployment + (provisEnd - provisStart);
        SuccessNumDeployment := SuccessNumDeployment +1;


     }
   }

 }
etry[] then
 { // case where the application is managed in ad-hoc way, with processes that might vary


   try [probLossApprovalRequest_AdHoc_Provisioning ==1] then
    {
      // loss Approval request. Nothing happens
      // This can really get unnoticed, as the user has left

      leaveNumLossApprovalRequest := leaveNumLossApprovalRequest +1;
      leaveNumWrongAccess := leaveNumWrongAccess +1;

      // general counter
      NumLossApprovalRequest := NumLossApprovalRequest +1;
      NumLossAHProvisioningApprovalRequest := NumLossAHProvisioningApprovalRequest +1;
```

```
      NumWrongAccess := NumWrongAccess +1;

    }
   etry [] then
    {
     // Approval happens.
     ApprovalReqStart := DEMOS_TIME;
     hold(waitingTimeMgmtApproval_AdHoc_Provisioning);
     ApprovalReqEnd := DEMOS_TIME;
     leaveOverallTimeApproval := leaveOverallTimeApproval + (ApprovalReqEnd - ApprovalReqStart);
     leaveSuccessNumApprovalRequest := leaveSuccessNumApprovalRequest +1;

     // general counter
     OverallTimeApproval := OverallTimeApproval + (ApprovalReqEnd - ApprovalReqStart);
     SuccessNumApprovalRequest := SuccessNumApprovalRequest +1;


     // Proceeding with de-provisioning (deployment) phase

     try [probLossExecutionActivity_AdHoc_Provisioning ==1] then
      {
       // probability of loss of configuration deployment/actual provisioning phase
       leaveNumLossDeployment := leaveNumLossDeployment +1;
       leaveNumWrongAccess := leaveNumWrongAccess +1;

       // general counter
       NumLossDeployment := NumLossDeployment +1;
       NumWrongAccess := NumWrongAccess +1;


      }
     etry [] then
      {

        // provisioning phase/deployment happens
        provisStart := DEMOS_TIME;
        hold(ConfigDeploymentTime_AdHoc_Provisioning);
        provisEnd := DEMOS_TIME;
        leaveOverallTimeDeployment := leaveOverallTimeDeployment + (provisEnd - provisStart);
        leaveSuccessNumDeployment := leaveSuccessNumDeployment +1;

        // general counter
        OverallTimeDeployment := OverallTimeDeployment + (provisEnd - provisStart);
        SuccessNumDeployment := SuccessNumDeployment +1;


       }
    }

  }

// general counters
```

```
  OngoingProvisActivities := OngoingProvisActivities -1;


}


//*****************************
// USER CHANGING ROLE/POSITION
//*****************************


class eventChangeGenerator =
{
  local var userNum = -1;
    repeat {

        // number of user changing role per Period Time
        userNum := rnd(numUserChangePerPeriodTime);
        //trace ("numUserChangePerDay=%v", userNum);

        do userNum
          {
            //trace ("EVENT USER Changing");
            entity(userChangeProcess, userChangeProcess, 0);
          }

        hold(weeks);
    }
}


// Process related to managing a new user changing role/position the company
class userChangeProcess =
{
  local var userRole = p_userRole;
  local var numTier1RequiredApps = p_numTier1ReqApps[userRole];
  local var numTier2RequiredApps = p_numTier2ReqApps[userRole];
  local var userLocation = p_userLocation;
  local var appRegion = -1;
  local var IAMProvisioningEnabled = -1;
  local var provActivity = -1;

  changeNum := changeNum +1;


  // managing process to get access to tier1 applications
  do numTier1RequiredApps
   {
     IAMProvisioningEnabled :=  p_AppWithCentralIAMProvisioning[TIER1_APP];
     appRegion := p_appManagementLocation;

     // counting effort, in terms of IAM automated and ad-hoc provisioning activities
```

```
    try [IAMProvisioningEnabled == 1] then
     {
       IAM_automation_effort:= IAM_automation_effort+1;
     }
    etry[] then
     {
       AH_effort := AH_effort +1;
     }


    provActivity := p_UserChange_ProvisActivity_PerApplication; // changing role might also require leaving and joining applications

    try [provActivity == USER_CHANGE] then
     {
        changeNumApp := changeNumApp+1;
        entity(userChangeProvisioningManagement, userChangeProvisioningManagement(TIER1_APP,#IAMProvisioningEnabled, #appRegion, #userRole, #userLocation), 0);
     }
    etry [provActivity == USER_JOIN] then
     {
        joinNumApp := joinNumApp+1;
        entity(userJoinProvisioningManagement, userJoinProvisioningManagement(TIER1_APP,#IAMProvisioningEnabled, #appRegion, #userRole, #userLocation), 0);

     }
    etry [provActivity == USER_LEAVE] then
     {
        leaveNumApp := leaveNumApp+1;
        entity(userLeaveProvisioningManagement, userLeaveProvisioningManagement(TIER1_APP,#IAMProvisioningEnabled, #appRegion, #userRole, #userLocation), 0);

     }
    etry [] then
     {
      //This should not really happen ...
     }


 }

  // managing process to get access to tier2 applications
do numTier2RequiredApps
 {
    IAMProvisioningEnabled := p_AppWithCentralIAMProvisioning[TIER2_APP];
    appRegion := p_appManagementLocation;

    // counting effort, in terms of IAM automated and ad-hoc provisioning activities
    try [IAMProvisioningEnabled == 1] then
     {
       IAM_automation_effort:= IAM_automation_effort+1;
     }
    etry[] then
     {
       AH_effort := AH_effort +1;
     }
```

```
    provActivity := p_UserChange_ProvisActivity_PerApplication;

    try [provActivity == USER_CHANGE] then
     {
       changeNumApp := changeNumApp+1;
       entity(userChangeProvisioningManagement, userChangeProvisioningManagement(TIER2_APP,#IAMProvisioningEnabled, #appRegion, #userRole, #userLocation), 0);
     }
    etry [provActivity == USER_JOIN] then
     {
       joinNumApp := joinNumApp+1;
       entity(userJoinProvisioningManagement, userJoinProvisioningManagement(TIER2_APP,#IAMProvisioningEnabled, #appRegion, #userRole, #userLocation), 0);

     }
    etry [provActivity == USER_LEAVE] then
     {
       leaveNumApp := leaveNumApp+1;
       entity(userLeaveProvisioningManagement, userLeaveProvisioningManagement(TIER2_APP,#IAMProvisioningEnabled, #appRegion, #userRole, #userLocation), 0);

     }
    etry [] then
     {
      //This should not really happen ...
     }

  }

}


// User Changing Role/Position - Management of Provisioning per Application

class userChangeProvisioningManagement (aType, emEnabl, aReg, uRole, uLocation) =
{
  local var appType = aType;
  local var userRole = uRole;
  local var userLocation = uLocation;

  local var IAMProvisioningEnabled = emEnabl;
  local var appRegion = aReg;

  local var ApprovalReqStart =0;
  local var ApprovalReqEnd = 0;

  local var provisStart =0;
  local var provisEnd = 0;
  local var carryOn =0;     // 0: do not carry on
                            // 1: carry on

  // dynamically calculated probability of bypassing Approval processes - dependency on failure rate
  local var probBypassApprovalProcess_IAM_AutomatedProvisioning = 0;
```

```
local var probBypassApprovalProcess_AdHoc_Provisioning = 0;


//trace ("USER CHANGING - APP TYPE=%v IAM Provisioning Enabled=%v  ", appType,IAMProvisioningEnabled );

// Starting the provisioning process

changeNumApprovalRequest := changeNumApprovalRequest + 1;

// general counters
NumApprovalRequest := NumApprovalRequest + 1;
OngoingProvisActivities := OngoingProvisActivities +1;


try [IAMProvisioningEnabled == 1] then
  { // case where the application is centrally managed, with the same predefined processes

    try [probLossApprovalRequest_IAM_AutomatedProvisioning ==1] then
     {

      // However ... Probability of bypassing the system (probBypassApprovalProcess_IAM_AutomatedProvisioning)

      probBypassApprovalProcess_IAM_AutomatedProvisioning := uniDist;


      try [probBypassApprovalProcess_IAM_AutomatedProvisioning < 1 -(1/(NumLossIAMProvisioningApprovalRequest+1)) ] then
        {
          // This is automated Provisioning Management. However, the user might still bypass the Approval process and directly
          // ask the sys admin to get their access rights to the application ...

          carryOn :=1;
          changeCarryOnDespiteNoApproval  := changeCarryOnDespiteNoApproval + 1;

          // there is a loss of Approval but the process is bypassed and carries on

          changeNumLossApprovalRequest := changeNumLossApprovalRequest +1;

          NumLossApprovalRequest := NumLossApprovalRequest +1;
          NumLossIAMProvisioningApprovalRequest := NumLossIAMProvisioningApprovalRequest +1;

          // general counters
          CarryOnDespiteNoApproval  := CarryOnDespiteNoApproval + 1;

        }
      etry[] then
        {
          // loss Approval request. Nothing happens
          changeNumLossApprovalRequest := changeNumLossApprovalRequest +1;
          changeNumMisconfigAccess := changeNumMisconfigAccess +1;  //no changes implies mis-configuration

          // general counters
          NumLossApprovalRequest := NumLossApprovalRequest +1;
```

```
      NumLossIAMProvisioningApprovalRequest := NumLossIAMProvisioningApprovalRequest +1;
      NumMisconfigAccess := NumMisconfigAccess +1;  //no changes implies mis-configuration

    }

 }
etry [] then
 {

  // Approval happens.

  carryOn := 1;

  ApprovalReqStart := DEMOS_TIME;
  hold(waitingTimeMgmtApproval_IAM_AutomatedProvisioning);
  ApprovalReqEnd := DEMOS_TIME;
  changeOverallTimeApproval := changeOverallTimeApproval + (ApprovalReqEnd - ApprovalReqStart);
  changeSuccessNumApprovalRequest := changeSuccessNumApprovalRequest +1;

  // general counters
  OverallTimeApproval := OverallTimeApproval + (ApprovalReqEnd - ApprovalReqStart);
  SuccessNumApprovalRequest := SuccessNumApprovalRequest +1;

 }

try [carryOn ==1] then
 {
  // Approval happened or were bypassed

  // Proceeding with provisioning (deployment) phase

  try [probLossExecutionActivity_IAM_AutomatedProvisioning ==1] then
   {
    // probability of loss of configuration deployment/actual provisioning phase
    changeNumLossDeployment := changeNumLossDeployment +1;
    changeNumMisconfigAccess := changeNumMisconfigAccess +1;  //no changes implies mis-configuration

    // general counter
    NumLossDeployment := NumLossDeployment +1;
    NumMisconfigAccess := NumMisconfigAccess +1;  //no changes implies mis-configuration


   }
  etry [] then
   {

     // provisioning phase/deployment happens
     provisStart := DEMOS_TIME;
     hold(ConfigDeploymentTime_IAM_AutomatedProvisioning);
     provisEnd := DEMOS_TIME;
     changeOverallTimeDeployment := changeOverallTimeDeployment + (provisEnd - provisStart);
     changeSuccessNumDeployment := changeSuccessNumDeployment +1;
```

```
        // general counters
        OverallTimeDeployment := OverallTimeDeployment + (provisEnd - provisStart);
        SuccessNumDeployment := SuccessNumDeployment +1;


        // Has the user account been misconfigured? (incorrect access control settings)
        try [probMisconfiguration_IAM_AutomatedProvisioning ==1] then
        {
          // user account has been misconfigured
          changeNumMisconfigAccess := changeNumMisconfigAccess +1;

          // general counters
          NumMisconfigAccess := NumMisconfigAccess +1;
        }
        etry[] then
        {
          // user account properly configured
        }
      }
    }
   etry [] then
    {
      // no approval and no bypassed processes
    }


  }
 etry[] then
  { // case where the application is managed in ad-hoc way, with processes that might vary

    try [probLossApprovalRequest_AdHoc_Provisioning ==1] then
     {

      // However ... Probability of bypassing the system (probBypassApprovalProcess_AdHoc_Provisioning)

      //probBypassApprovalProcess_AdHoc_Provisioning := binom(1, 1 -(1/(NumLossAHProvisioningApprovalRequest+1)) ) ;


      probBypassApprovalProcess_AdHoc_Provisioning := uniDist;


      try [probBypassApprovalProcess_AdHoc_Provisioning < 1 -(1/(NumLossAHProvisioningApprovalRequest+1))] then
        {
          // This is AD-HOC Management. The user might bypass the Approval process and directly
          // ask the sys admin to get their access rights to the application ...
          carryOn :=1;
          changeCarryOnDespiteNoApproval  := changeCarryOnDespiteNoApproval + 1;

          // there is a loss of Approval but the process is bypassed and carries on

          changeNumLossApprovalRequest := changeNumLossApprovalRequest +1;
```

```
      NumLossApprovalRequest := NumLossApprovalRequest +1;
      NumLossAHProvisioningApprovalRequest := NumLossAHProvisioningApprovalRequest +1;

      // general counters
      CarryOnDespiteNoApproval  := CarryOnDespiteNoApproval + 1;

    }
   etry[] then
    {
      // loss Approval request. Nothing happens
      changeNumLossApprovalRequest := changeNumLossApprovalRequest +1;
      changeNumMisconfigAccess := changeNumMisconfigAccess +1;  //no changes implies mis-configuration

      // general counters
      NumLossApprovalRequest := NumLossApprovalRequest +1;
      NumLossAHProvisioningApprovalRequest := NumLossAHProvisioningApprovalRequest +1;
      NumMisconfigAccess := NumMisconfigAccess +1;  //no changes implies mis-configuration

    }

  }
 etry[] then
  {
    carryOn := 1;

   // Approval has been granted

   ApprovalReqStart := DEMOS_TIME;
   hold(waitingTimeMgmtApproval_AdHoc_Provisioning);
   ApprovalReqEnd := DEMOS_TIME;
   changeOverallTimeApproval := changeOverallTimeApproval + (ApprovalReqEnd - ApprovalReqStart);
   changeSuccessNumApprovalRequest := changeSuccessNumApprovalRequest +1;

   // general counters
   OverallTimeApproval := OverallTimeApproval + (ApprovalReqEnd - ApprovalReqStart);
   SuccessNumApprovalRequest := SuccessNumApprovalRequest +1;
  }


try [carryOn ==1] then
 {
   // Approval has been granted or it might have been bypassed

   // Proceeding with provisioning (deployment) phase

   try [probLossExecutionActivity_AdHoc_Provisioning ==1] then
    {
      // probability of loss of configuration deployment/actual provisioning phase
      changeNumLossDeployment := changeNumLossDeployment +1;
      changeNumMisconfigAccess := changeNumMisconfigAccess +1;  //no changes implies mis-configuration
```

```
          // general counters
          NumLossDeployment := NumLossDeployment +1;
          NumMisconfigAccess := NumMisconfigAccess +1;  //no changes implies mis-configuration


        }
       etry [] then
        {

          // provisioning phase/deployment happens
          provisStart := DEMOS_TIME;
          hold(ConfigDeploymentTime_AdHoc_Provisioning);
          provisEnd := DEMOS_TIME;
          changeOverallTimeDeployment := changeOverallTimeDeployment + (provisEnd - provisStart);
          changeSuccessNumDeployment := changeSuccessNumDeployment +1;

          // general counters
          OverallTimeDeployment := OverallTimeDeployment + (provisEnd - provisStart);
          SuccessNumDeployment := SuccessNumDeployment +1;


          // Has the user account eventually been  misconfigured? (incorrect access control settings)
          try [probMisconfiguration_AdHoc_Provisioning ==1] then
          {
            // user account has been misconfigured
            changeNumMisconfigAccess := changeNumMisconfigAccess +1;

            // general counters
            NumMisconfigAccess := NumMisconfigAccess +1;

          }
          etry[] then
          {
            // user account properly configured
          }
        }
      }
    etry [] then
     {
       // no approval and no bypassed processes
     }

   }

  // general counters
  OngoingProvisActivities := OngoingProvisActivities -1;

}



// class keeping track of variables of relevance of this model
```

```
class measurement =
{
   repeat {

      // Calculating Access Accuracy measure
      try[NumApprovalRequest>0] then
       {

         Access_accuracy := 1 - ((access_accuracy_UAD_weight* NumDeniedGoodAccess +
                              access_accuracy_UAM_weight* NumMisconfigAccess +
                              access_accuracy_UAH_weight* NumWrongAccess))/NumApprovalRequest;

       }
      etry[] then
       {
         // initial setting of accuracy = 0
       }


      // Calculating Approval Accuracy   - SuccessNumApprovalRequest + CarryOnDespiteNoApproval
      try [ SuccessNumApprovalRequest + CarryOnDespiteNoApproval >0] then
        {
         Approval_accuracy := SuccessNumApprovalRequest/(SuccessNumApprovalRequest+CarryOnDespiteNoApproval);
        }
      etry [] then
        {
         // initial setting of accuracy = 0
        }


      // Productivity Cost - Pessimistic estimate (waiting times could be double counted - as eventually there could be a provisioning failure)

      Productivity_cost := unit_cost_per_day * (joinOverallTimeApproval + joinOverallTimeDeployment + changeOverallTimeApproval + changeOverallTimeDeployment) +
                        unit_cost_failure * (joinNumLossApprovalRequest + joinNumLossDeployment + changeNumLossApprovalRequest + changeNumLossDeployment);


      // IAM - Provisioning Automation cost

      try [numAppsWithCentralIAMProvisioning[TIER1_APP] + numAppsWithCentralIAMProvisioning[TIER2_APP] >0] then
       {
         IAM_automation_cost := IAM_provisioning_fixed_cost + (numAppsWithCentralIAMProvisioning[TIER1_APP] + numAppsWithCentralIAMProvisioning[TIER2_APP] )*IAM_provisioning_variable_cost;

       }
      etry [] then
       {
          // no IAM Provisioning automation at all. No costs
       }



      trace ("demos_sample_tick=%v", demos_sample_tick);
      trace ("join_numUser=%v", joinNum);
```

```
trace ("join_numApp=%v", joinNumApp);

(* trace ("join_NumApprovalRequest=%v", joinNumApprovalRequest );
trace ("join_NumLossApprovalRequest=%v", joinNumLossApprovalRequest);
trace ("join_OverallTimeApproval=%v", joinOverallTimeApproval);
trace ("join_successNumApprovalRequest=%v", joinSuccessNumApprovalRequest );
trace ("join_NumLossDeployment=%v", joinNumLossDeployment );
trace ("join_OverallTimeDeployment=%v", joinOverallTimeDeployment );
trace ("join_successNumDeployment=%v", joinSuccessNumDeployment );
trace ("join_numMisconfigAccess=%v", joinNumMisconfigAccess );
trace ("join_numDeniedGoodAccess=%v", joinNumDeniedGoodAccess );
trace ("join_CarryOnDespiteNoApproval=%v", joinCarryOnDespiteNoApproval ); *)

trace ("leave_numUser=%v", leaveNum);
trace ("leave_numApp=%v", leaveNumApp);
(*trace ("leave_NumApprovalRequest=%v", leaveNumApprovalRequest );
trace ("leave_NumLossApprovalRequest=%v", leaveNumLossApprovalRequest);
trace ("leave_OverallTimeApproval=%v", leaveOverallTimeApproval);
trace ("leave_successNumApprovalRequest=%v", leaveSuccessNumApprovalRequest );
trace ("leave_NumLossDeployment=%v", leaveNumLossDeployment );
trace ("leave_OverallTimeDeployment=%v", leaveOverallTimeDeployment );
trace ("leave_successNumDeployment=%v", leaveSuccessNumDeployment );
trace ("leave_NumWrongAccess=%v", leaveNumWrongAccess); *)

trace ("change_numUser=%v", changeNum);
trace ("change_numApp=%v", changeNumApp);
(*trace ("change_NumApprovalRequest=%v", changeNumApprovalRequest );
trace ("change_NumLossApprovalRequest=%v", changeNumLossApprovalRequest);
trace ("change_OverallTimeApproval=%v", changeOverallTimeApproval);
trace ("change_successNumApprovalRequest=%v", changeSuccessNumApprovalRequest );
trace ("change_NumLossDeployment=%v", changeNumLossDeployment );
trace ("change_OverallTimeDeployment=%v", changeOverallTimeDeployment );
trace ("change_successNumDeployment=%v", changeSuccessNumDeployment );
trace ("change_numMisconfigAccess=%v", changeNumMisconfigAccess );
trace ("change_numDeniedGoodAccess=%v", changeNumDeniedGoodAccess );
trace ("change_CarryOnDespiteNoApproval=%v", changeCarryOnDespiteNoApproval ); *)

trace ("NumApprovalRequest=%v", NumApprovalRequest );
trace ("NumLossIAMProvisioningApprovalRequest=%v", NumLossIAMProvisioningApprovalRequest);
trace ("NumLossAHProvisioningApprovalRequest=%v", NumLossAHProvisioningApprovalRequest);
trace ("NumLossApprovalRequest=%v", NumLossApprovalRequest);
trace ("CarryOnDespiteNoApproval=%v", CarryOnDespiteNoApproval );
trace ("successNumApprovalRequest=%v", SuccessNumApprovalRequest );
trace ("OverallTimeApproval=%v", OverallTimeApproval);
trace ("NumLossDeployment=%v", NumLossDeployment );
trace ("OverallTimeDeployment=%v", OverallTimeDeployment );
trace ("successNumDeployment=%v", SuccessNumDeployment );
trace ("numMisconfigAccess=%v", NumMisconfigAccess );
trace ("numDeniedGoodAccess=%v", NumDeniedGoodAccess );
trace ("NumWrongAccess=%v", NumWrongAccess);
trace ("OngoingProvisActivities=%v", OngoingProvisActivities);
```

```
        //Aggregated measures
        trace ("Access_accuracy=%v", Access_accuracy);
        trace ("Approval_accuracy=%v", Approval_accuracy);
        trace ("Productivity_cost=%v", Productivity_cost);
        trace ("IAM_automation_cost=%v", IAM_automation_cost);
        trace ("IAM_automation_effort=%v", IAM_automation_effort);
        trace ("AH_workload=%v", AH_effort);

        demos_sample_tick :=  demos_sample_tick + 1;
        hold(observedTimePeriod);
    }
}


// initialising simulation
entity(initialise, initialise,0);


// holding for the entire simulation timeframe
req [done ==1];

close;
```