Using assurance models to aid the risk and governance lifecycle♦

Adrian Baldwin, Yolanta Beres, Simon Shiu
Trusted Systems Laboratory
HP Laboratories Bristol
HPL-2007-48
March 28, 2007*

trust, assurance, risk, compliance, governance, security

In this paper we describe an enterprise assurance model allowing many layers of the enterprise architecture from the business processes; supporting applications and the IT infrastructure and operational processes to be represented and related from a control and risk perspective. This provides a consistent way of capturing and relating the risk views for the various stakeholders within the organisation. At the low-level we use assurance models to provide automated testing of controls and policies and at the higher level these results are related across the enterprise architecture. This enables a repository for manual and automated test results that can be used to derive different (but consistent) views for the various stakeholders.

* Internal Accession Date Only
♦ BT Technology Journal, Vol. 25, no. 1, Jan. 07

Approved for External Publication

© Copyright 2007 British Telecommunications plc

Using assurance models to aid the risk and governance lifecycle.

Adrian Baldwin, Yolanta Beres, Simon Shiu
HP Labs Bristol

Abstract

In this paper we describe an enterprise assurance model allowing many layers of the enterprise architecture from the business processes; supporting applications and the IT infrastructure and operational processes to be represented and related from a control and risk perspective. This provides a consistent way of capturing and relating the risk views for the various stakeholders within the organisation. At the low-level we use assurance models to provide automated testing of controls and policies and at the higher level these results are related across the enterprise architecture. This enables a repository for manual and automated test results that can be used to derive different (but consistent) views for the various stakeholders.

1 Introduction

Enterprise IT systems are becoming increasingly complex yet at the same time businesses are more and more reliant on them. A company and its executives need to ensure that operational and IT risks are understood, that appropriate policies and controls are in place to mitigate these risks, that these controls are being used properly; and that they are effective in mitigating risks. Within an enterprise there are many people involved in the risk management life-cycle including those designing and running business processes; those implementing and running applications; the security office setting policy; Internal audit maintaining and checking on the control framework; and the IT operations staff who need to follow all the policies and processes and demonstrate necessary controls are being maintained.

This paper describes how an assurance model that represents the control architecture and its relationship to the enterprise architecture can be used to support enterprise risk management. Our approach changes a manual documentation based process into one using a structured assurance model allowing for more consistency, efficiency and in some areas automation. The paper focuses mainly on assurance management, as a part of risk management. We show how the sharing of model based assurance views amongst the various stakeholders leads to cultural change where risks are continuously managed rather then just when auditors appear. The approach has been piloted within two companies demonstrating that it can help to automate audits and produce risk based reports useful both to auditors and application owners in gaining a better handle on risk.

We start this paper by looking at enterprise computing from a perspective of supporting the critical business processes and the risks that may occur. Section 3 then describes an assurance modelling framework we have developed to provide assurance that risks are being appropriately managed. We describe how we have automated and improved aspects of an audit programme and then show how we can build on this to provide a set of views on risk to the various stakeholders within the enterprise. Section 4 provides a wider discussion of related work and the contribution to enterprise risk management. Section 5 provides a final summary and conclusion.

2 Enterprise Architecture and Risk

This section starts with an overview of enterprise architecture, which is a standard practice for ensuring IT is well managed. This helps to define the overall set of systems and processes where operational risk will occur within the enterprise, and hence, where assurance is needed.

Finally, we discuss the enterprise risk lifecycle which provides the context for the model based approach described in section 3.

2.1 Enterprise Architecture

An enterprise architecture [1][2][3] describes the business processes, organisation and information systems as they exist or should exist and as such provides an overarching framework defining how IT investments should be made and how the IT systems should be run. Typically, enterprise architectures follow the layers shown in Figure 1. There is a business process layer identifying and documenting the major business processes on which the business relies. This will, for example, include financial processes, order management, human resources, sales, marketing product support, supply change management, asset management, planning, product creation.

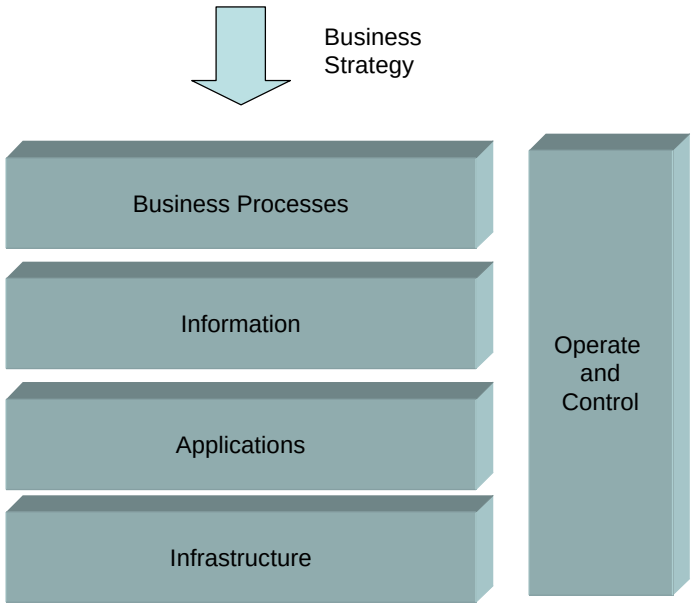

Figure 1:
The Layers within an enterprise architecture

The information layer defines how information assets should be managed and helps establish how these information assets can be used to provide better business intelligence. This layer includes data modelling, messaging tools, meta-data descriptions and data handling principals along with data warehousing tools.

The application layer describes the various applications and services that directly help in performing the business processes defined in the top layer. This would include ERP systems that support the various supply chain and financial processes.

The infrastructure layer includes most of the IT including database systems, servers, networking as well as components such as directory services, single sign on, change management systems and identity provisioning systems.

The enterprise architecture identifies the key stakeholders at each layer and for each element within the architecture. For example, business owners who are responsible for the smooth running of the business processes; the application owners who are responsible for managing the applications in support of the business process and various people responsible for different parts of the infrastructure.

The enterprise architecture provides a set of principles and guidelines by which the company achieves its business goals and a set of policies and standards describing how things should, or shouldn't be achieved, for example, ensuring that IT changes meet changing business needs.

The operate and control box includes all those tasks that makes sure that the systems are run in an effective and controlled manner. Typically, IT will be managed through a set of processes derived from the ITIL standards, see [7],[8]. These help ensure IT systems and applications are run to support the business and that there are planning processes to ensure appropriate levels of service are maintained.

2.2 Risk in the enterprise architecture

Within this paper we are interested in risks that have a significant business impact either through a direct loss of money or assets or through the inability to effectively operate business processes. Primarily we are looking at how operational risks can be controlled and how the various stakeholders identified within the enterprise architecture can gain assurance that risks they are responsible for, or those that they are reliant on are under control.

Risks and mitigations can appear at all levels within the enterprise architecture, for example:

- At the business process level:
    - an individual may be able to initiate and validate a transaction
- At the application level:
    - the wrong person may gain access to a given process via the supporting application;
    - or a risk may be mitigated because an application doesn't allow a given transaction;
- At the infrastructure level:
    - there may be a risk that business data can be changed by direct access to a database, through root access to the server or access to a disk on a SAN.

Moreover, risks will often depend on inter-relationships between the levels. As such it is important to understand risk in the context of the enterprise architecture, and to be able to navigate between the layers.

2.3 The enterprise risk lifecycle

Managing risk between so many stakeholders and across such a complex architecture is clearly a challenge. Figure 2 shows a life-cycle that goes from the initial risk assessment through implementation of mitigation strategies, operation of the environment and controls, and finally audit and review to ensure the correct risk profile is being maintained.

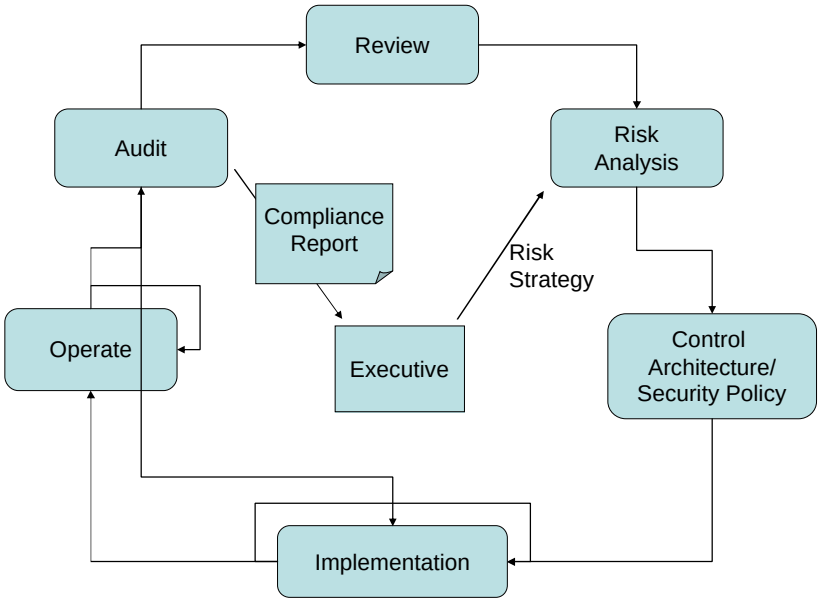

Figure 2:
The enterprise risk lifecycle

Risk Analysis involves the overall assessment of how risk will impact on the business. To understand this, business assets and missions need to be identified along with the impact of loss of an asset or failure of a mission. At this level broad categories of threats are considered and the probability of them happening is considered to derive a risk calculation. This risk can be considered against the risk strategy or appetite set by the board and as such risks that need mitigating are identified. There are a number techniques and tools that help in performing an enterprise risk assessment [4].

The risk analysis then feeds into the control architecture and security policy design work where mitigations are designed. This consists of a number of control objectives which are *'statements of the desired results of implementing a control procedure for a particular activity'* and are often described along with the risk that they are mitigating. Governance frameworks, such as COBIT [5], provide a reference set of *C*ontrol *OBj*ectives for *I*nformation and related *T*echnology. The control objectives may also be driven by the need for legal compliance – for example, the Sarbanes Oxley (SOX) act [6] has driven an increase in corporate controls and helped drive the adoption of the COBIT governance framework. Often the security office will produce a set of policies and procedures that determine how these control objectives are going to be met.

At all levels, as solutions are implemented and deployed the stakeholders must demonstrate that they are meeting a given set of local controls that achieve the control objectives. In these terms a control is defined as *'policies, procedures, practices and organisational structure designed to provide reasonable assurance that business objectives will be achieved and that undesired events will be prevented or detected and corrected'*. Processes implementing the local controls will normally be designed by the system owner with an Internal Audit department endorsing that the controls meet the control objectives and testing that they are run properly. An example of the audit process and example controls is covered in more detail in section 3.2.1.

Example controls may relate to checks that need to be performed in a business process through to checks on data centre door lock configurations. It is the complete set of these controls at the various layers that ensures risk is mitigated appropriately.

Once a solution is developed it is passed to an operational environment where it will be run with best practice processes and procedures such as those defined within ITIL [7][8]. The implementation phase includes the design and creation of an appropriate operational environment which implements many of the controls and is constrained by many of the security policies. We then have an operational process where the IT operations staff are responsible for the correct running of the processes including all the controls.

The audit function has the responsibility of checking that the business is running in line with the control objectives set to both mitigate risk and meet legal requirements. Hence, they have their own testing cycle where an Internal Audit department (reporting to the CFO) will sample and report on various controls. Audit produces reports for the stakeholders being audited, the responsible executives and the CFO who is responsible for compliance. Additional independent external audit will also take place where regulation demands and much of the internal audit effort is in place to reduce the external audit burden and costs.

The lifecycle is a closed loop in that, issues found will lead to change in the control architecture. In addition business and environmental changes imply that the risk analysis should be regularly reviewed. It is also the case that stakeholders responsible for significant entities will regularly review control objectives for their section of the enterprise architecture.

3 Assurance Modelling

Much of the risk management lifecycle is a manual process with policies being defined on web pages; policy compliance being checked by self assessment questionnaires and review. Audit controls are often maintained within hundreds of excel spreadsheets with at best document management systems being used to maintain audit results. The general research challenge we are addressing is how to use technology to improve the rigor and efficiency of this people based lifecycle. Our approach is to model the control architecture and its relationship to the enterprise architecture. The model provides a consistent repository and allows for automation. In this section we focus on assurance management, which includes the audit function described above, but also involves providing meaningful information to many other stakeholders, e.g. Chief Information Security Officer (CISO), Chief Risk Officer, Business owners, application and infrastructure managers and so on.

Different stakeholders tend to have different approaches and concerns, for example auditors mostly emphasize process controls and will actually sample workflow data from the environment, whereas security experts tend to use manual questionnaires and software tools to check that the state and configuration of IT systems is in line with policies, or mitigating known threats. More senior figures, such as the director of internal audit or the CISO, are more interested in aggregated views, perhaps related to the enterprise or control architecture, and perhaps key risk indicators (KRIs) that hint at emerging risk, see [9]. KRIs are often indirect measures that can easily be benchmarked and trended to help identify where, say, limited internal audit resources should pay attention. For example, monitoring the number of dormant accounts is a lightweight and indirect way of tracking whether account administration is maintaining appropriate control.

We have developed a modelling approach that captures these different kinds of assurance information and allows them to be combined in different ways for the different stakeholders. In this section we provide an overview of the modelling framework and then illustrate in detail how it has been applied to support the assurance reporting needs of internal auditors. Then we describe how the same framework can be used to create views that cut across the enterprise architecture in ways that support some of the other assurance stakeholders.

3.1 The Assurance Modelling Architecture

Our approach of modelling allows control information to be captured in a structured form that relates elements across the enterprise architecture with risks and associated risk mitigation

strategies. This approach allows us to build assurance tools to support a better understanding of risk within the layers of the enterprise architecture. Here we describe a tool set for creating an assurance model and performing the automated testing and reporting.

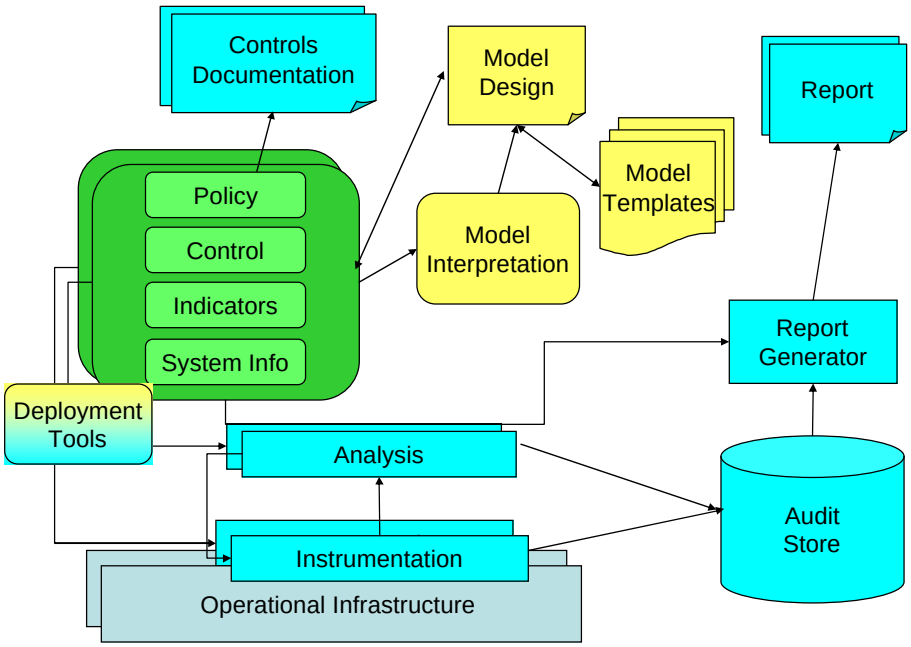

Figure 3:
The Assurance Modelling architecture

Figure 3 shows the architecture for the assurance modelling systems. A mature enterprise will have a set of control objectives, controls and policies already defined as textual documents. Modelling such a control framework involves capturing this information along with how this relates to the enterprise architecture. A less mature enterprise could use a set of templates that, for example, follow from COBIT. A variety of templates would be needed that express control objectives for different elements within the enterprise architecture. For example, there would be a control template for applications, servers, databases and for a variety of standard business processes. Such templates would need customising but provide a catalogue of risk concerns thereby helping kick start a risk management programme.

In addition to relating the controls, the model captures the different ways stakeholders want to test for emerging risk or effectiveness of controls. The modelling tools cover the kind of control/process tests that auditors perform, KRI's, and state and configurations tests. In this way different stakeholders can create unique views with combinations of the above assurance information.

Once defined a model can then be deployed onto systems (assuming that suitable instrumentation is available) and the model automates much of the routine audit test work. This is described in more detail in section 3.2. The model automatically creates web based compliance reports for individual areas of concern such as applications, servers, or data centre processes. These reports are hierarchical showing an overall traffic light based status at the top and allowing those viewing the reports to dig down into detailed compliance test results.

A separate set of assurance models can be created that pull together the results of the automated testing and produce different assurance views of the enterprise architecture. These views can also include results from manual audits, or other manual compliance testing. These meta-models allow the risk concerns of the various stakeholders within the enterprise

architecture to be captured and related to different risk concerns. The meta-models are created in the same way as the more detailed control models and can refer to results from other models.

Providing compliance views for the many stakeholders within the enterprise architecture not only helps in understanding the state of compliance for the enterprise but should lead to better risk management as the various system applications, and business owners now have a view on the risks they should be managing.

3.2 Automated Controls Testing

The discussion of the risk life-cycle defined concepts of control objectives and controls – typically these are tested by auditors who carry out test-work against each control. Much of the current audit cycle is a manual process. Once an audit is announced auditors will pick samples for each control they are going to test – for example, in testing user management they will pick a number of users. Auditors then require the owners of the system to provide them with evidence showing that the controls have been appropriately applied. A typical sample size may be 10 or 25 samples depending on the type of the audit. Audit tools such as *Audit Command Language* (ACL) exist to help an auditor compare and organise the sample data that they receive.

The assurance modelling allows the audit test-work to be represented within the model as a number of tests. An analysis engine can then take the model along with databases containing the evidence and produce a report showing how well the control architecture is being run. This report would also include risk indicators showing the effectiveness of risk management. The analysis engine works over all the available data rather than a small sample thus increasing the accuracy and reducing mistakes made in manual analysis. The structure of the controls testing model is described in section 3.2.2 with section 3.2.3 providing a description of the underlying representation and mechanisms supporting the tool.

3.2.1 Application Audit

We illustrate the way the assurance modelling applies to an audit by showing how it applies to an application audit. Firstly, we describe the application audit process. Auditors will typically audit applications supporting the critical business processes and particularly those supporting the financial processes, and hence, under the remit of the Sarbanes Oxley Act. The application audit considers the operational environment for the application, that is, how it is run, rather than looking at the business processes themselves. These are audited as part of a separate process although an integrated audit process would run audits over multiple levels concurrently to gain a better overview.

In a well run audit programme auditors have a standard control objective/control framework for operating an application. The lead auditor can review and adapt the template in light of the current risk environment. Even though the business process is not being directly audited the risks or control objectives should be at the level of events that affect the business. Each risk will have one or more controls that help mitigate the risk and each of these controls will have one or more pieces of test-work that are used to test the control. An example segment audit template is given in Figure 4.

Control Objective: Active User accounts should not exist for terminated employees

***Control*:** User and admin accounts are inactivated within 30 days of termination

Testwork: Check terminated users from enterprise directory.

Control Objective : Changes are adequately tested before going to production

Control: Acceptance test signoff performed in test environment

Testwork: Check samples of emergency change testing signoff

Testwork: Check samples of non-emergency change testing signoff

Control Objective : Users may not have unauthorized access to view or update data

Control: Authorisation matrix defines who can approve access to roles

Testwork : Check and review authorisation matrix

Control: Only authorised personnel have access to production databases

Testwork : Check database access privileges

Control: Requests for new and modified accounts are documented

Testwork : Check documentation for a sample of users

Figure 4:
Segment of an audit controls and test work script.

The example shows three control objectives that have been identified as important to the business; two concern well managed access and the third applies to risks associated with changes to an application. The testwork descriptions contain additional detail, for example, including sample sizes which would be decided on the basis of previous audits and audit guideline.

Often the testwork will start with the auditor having to understand the process by which things happen. For example, on looking at the signoff of tests the auditor will have to understand the process that the application owner uses; makes a judgement on whether the process is sufficient and then checks it is run correctly. This task is made simpler where consistent sets of tools and management processes are used throughout an IT organisation – however, the way IT has grown organically and where companies have grown through acquisitions there is often a range of management systems and styles used.

An auditor will then carry out analysis based on the scope of the testwork descriptions and the results of the testwork are then included in a report based on the audit template. Where issues are found in the audit these are then identified as audit issues and a response is expected from the application owner. Each issue is classified and coloured according to its severity, for example, serious (red), reportable (yellow), discussion points (green) and a recommendation is made. The audit report will be passed up the responsible management chain. The application owner must provide an action plan for rectifying the audit issues.

3.2.2 Automating Audit Testing

Within the audit process there is a considerable manual step associated with understanding the processes, gathering data and validating it. The model based assurance approach starts by relating the control framework to the particular situation. We have created a GUI based tool that allows the structure of the audit programme to be captured as shown in figure 5. Here the risks, controls and testwork goals are represented as nodes within a graph where the links represent controls mitigating risks or testwork that validates controls. In addition, the risks can be associated with various operational (e.g. ITIL) processes or tasks within these processes.

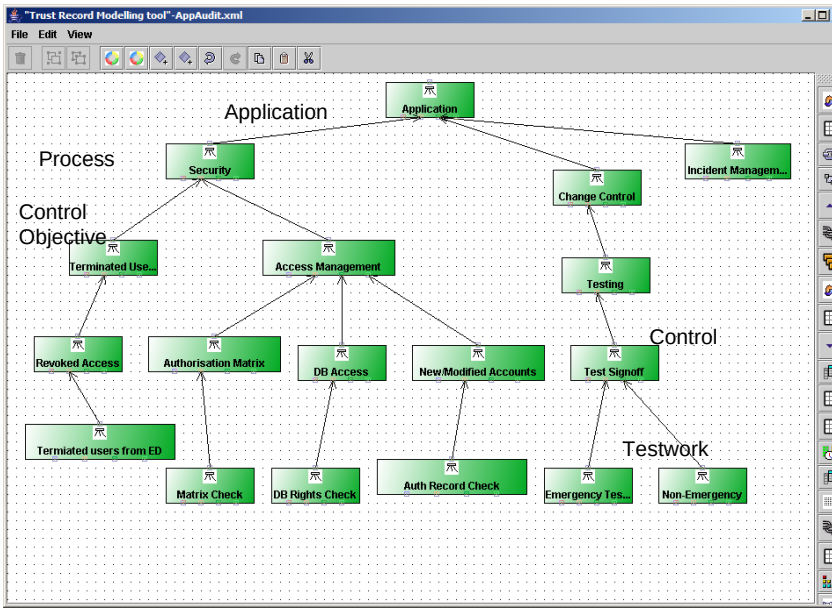

Figure 5:
Capturing the structure of the audit programme.

Each node has a descriptive element that captures the role it plays within the control framework (e.g. a process, control, control objective, risk, testwork) and according to the type a number of attributes are also represented. The top node of the graph represents the thing being audited or tested; in this case an application but it could be a server, database, data centre or business process. These attributes are used to capture the details of the audit programme that would typically be contained in a word document or an excel spread sheet. Capturing the audit process allows us to understand the relationships between the different pieces of testwork, controls and the processes within the operational environment.

To automate the audit testing we need to dig further into the management processes that occur and ensure that the checks, signoffs and other controls are enforced. Here the modelling framework has an extensible set of tests and data-comparison methods linked to a node such as those that would be used by an auditor. Here the graph represents a data-flow where at the bottom we have nodes representing data sources and nodes further up the graph represent tests carried out on the data. Each test can be configured with detailed parameters for the test and there is a mapping mechanism that maps between the available data and the inputs required for the test. Tests themselves produce data that can be used in further tests.

Figure 6 shows an example of linking a series of tests together to check that new roles are approved. Here there are three branches with a number of nodes that identify new roles within the SAP system and check these against a set of e-mail approval archives. The first branch identifies any users with new roles where there is no approval e-mail; the middle has some indicative metrics and the third identifies new roles with approvals and shows the approval e-mails. Note that for audit purposes it is necessary to show the e-mail or web form showing exactly what was approved and seen by the approver.

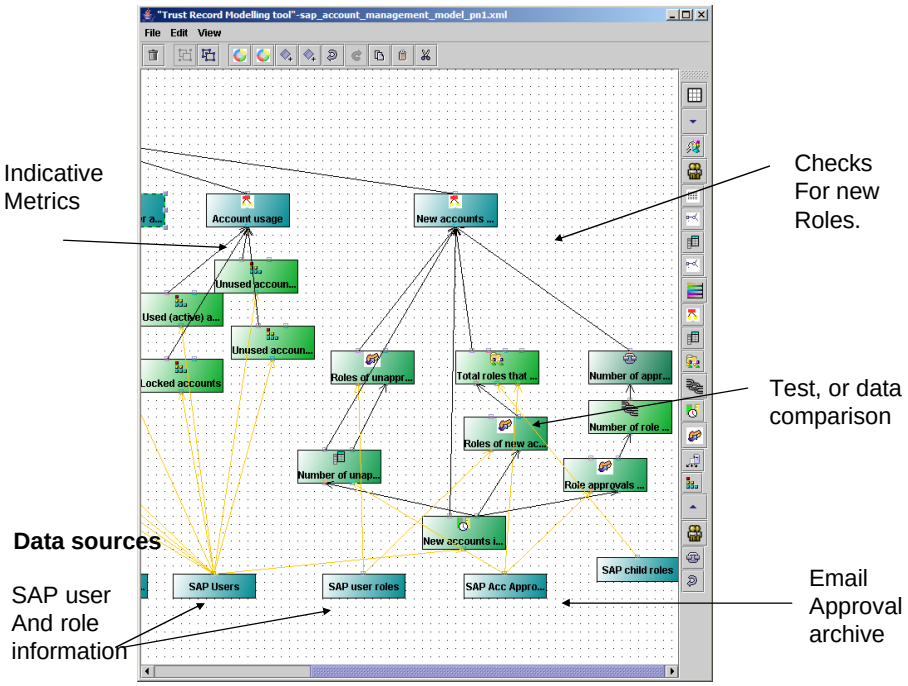

Figure 6:
Example tests on new user roles added to a SAP system.

The test results include detailed data as a result of analysis and often summary metrics which are then used to derive a status for the overall testwork. In this example the status of the testwork is based on the number of roles allocated with no approvals. There is a second branch shown in the figure that includes a number of metrics that are indicative of good role and account management, for example the number of unused active accounts or locked accounts. These metrics may indicate risk either due to a control not operating correctly or due to an ineffective control. Each metric included in the model is accompanied by a graph defining its acceptable and unacceptable values.

Having captured the structure of the audit programme and the details of the testwork in an assurance model we can now perform automated audit testing. Reports will be run at a regular interval; having first ensured data is pulled into the audit database. The reports follow the structure of the model, and hence, reflect the structure of the audit programme. The report includes a traffic light indicator showing the status for the overall audit.

The reports can be navigated according to the structure of the model and each of the risks, controls and testwork sections will have a status value. These can help drive the navigation so that an auditor or application owner can quickly find the areas where there are risks emerging or controls failing and drill down to the details of the testwork, for example, listing users without authorisation for their roles. The reports can also include information that can help an auditor produce the samples they need.

3.2.3 Model Structure

Underlying the GUI tool a given model is represented as an a-cyclic graph where each node can have a number of inputs and outputs depending on its type, and its relationship with other nodes. The GUI tool renders the model to an XML structure (for convenience of parsing) which has the form shown in Figure 7. The model consists of a set of nodes containing both a description and the details of an assurance test or combination rule. There are then a set of connectors defining the graph structure and associated data mappings.

```
<Model>
  <VariableSet>
          <VariableName/>*
 </VariableSet>
 <ContextSet>
    <Context name>
         <Variable name value/>*
   </Context>
 </ContextSet>
<NodeList>
     <node id name>*
          <description>
              <controlframeworktype id/>
              <attrib name value/>*
               <desc> text </desc>
          </description>
          <assuranceTest name>
                    -- test specific description
          </assuranceTest>
          <outputset>
              <output name type handler>*
     </node>
<NodeList>
```

```
<connectors>
   <connect sourcenode sourceport
              targetnode targetport>*
        <sources>
          <source name type constraint>*
        </sources>
       <mapping>
          <target name source>*
       </mapping>
  <connectors>
</Model>
```

Figure 7:
The overall structure of the assurance model.

Each node represents an entity within the control framework. The entities can be either concepts in the enterprise architecture whose risk profile is being described, or abstractions that describe how to test, measure or structure assurance information. Connections between the nodes represent relationships between these entities with the meaning being dependent on the input and output ports in the nodes as well as the entities being connected.

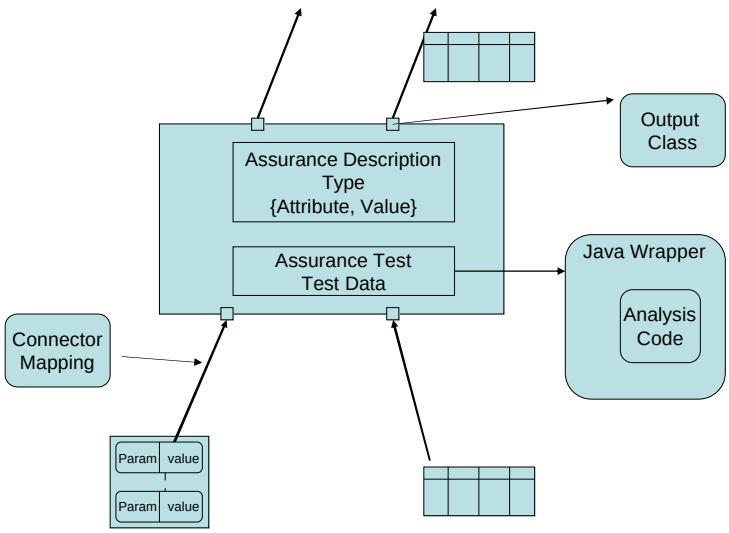

Figure 8:
The structure of a node within the assurance model.

Each node has a number of inputs and outputs; these describe the relationships between nodes and are also used to define data-flows during the analysis. The node has a descriptive element including a type and a set of (attribute, value) pairs that describe the thing that the node is modelling; this is described in more detail in section 3.3.1. The node also has an assurance

test defining the relationships between the inputs and outputs. Where the node has the role of performing audit tests, there is an available set of tests for comparing various sets of data, performing checks such as separation of duty, checking event orders etc. Where the assurance model is relating metrics, or the results from testing, the assurance test method takes the form of a rule specifying how the status from the various input relationships should be combined. Further test types, described later in section 3.3.2, allow results to be obtained from other assurance models.

The assurance modelling framework has been implemented in Java with the framework providing an object representation of the model, support for data handling, and report generation capabilities. The data handling within the framework abstracts away from the database where all audit data and analysis results are stored; data is automatically mapped into the form required by an assurance test and can be in the form of a table or a set of (attribute, value) pairs. We have implemented a number of assurance test types to process and compare data sets along with a set of combination rules. The framework supports the extension of this set by extending existing Java classes conforming to a given interface along with an XML description of the node structure.

Analysis results are stored within the database. The reports are generated by a number of servlets that read the model structure and render the analysis description according to the description of the node structure. Links are placed within the report according to the model structure allowing a user to navigate through the report.

3.2.4 Results

We have piloted the model based assurance approach with auditors within two companies to demonstrate how it can support automated audit and risk reporting. Here, we worked alongside auditors to produce models that follow their testwork and reporting requirements for application audits. Whilst as a prototype system they cannot rely on the data, we have had positive responses from both the application owners and the auditors. The immediate advantages of the approach come from the reduction in time taken to obtain data and the increased accuracy due to testing all data rather than just a sample. Once the tool is validated and trusted by auditors it could save a considerable amount of the routine audit work allowing auditors to spend more time investigating risk issues and enabling a greater coverage of enterprise applications.

We see longer term benefits from the sharing of risk information. The model provides the application owner a clear view on the audit program and why the controls are in place. Regular reports allow them to see that control is being maintained or where there may be issues. Trending between the reports allows them to see if they are improving or maintaining control. Having reports showing the application owners how well they are mitigating risk allows them to respond quickly to issues and maintain controls. This helps avoid a common situation where systems are often only compliant just before an audit or where the application owner gets a surprise from the periodic audit.

3.2.5 Template Reuse

Even after using a model based approach to assurance the costs of deploying an assurance tool onto an application could be considerable. This may involve a consultant who needs to gain an understanding of the control framework, the operational processes and how the testwork is carried out. This does, however, become a one off cost rather than the regular discovery process that happens each time an audit team arrive – it could even be done alongside a regular audit. Ideally, however, we would customise a set of standard models for a given customer.

Towards this goal, models can be created containing variables that can be bound either during the deployment phase or from the parent model during analysis and reporting. This allows model segments to be created for standardised processes (for example, user and change

management) that are tested for many entities within the control framework. It also allows the reuse of say an application model for multiple applications where details of a specific application are derived from a central source such as a Configuration Management Database (CMDB). Using a common set of templates reduces the job of modelling for each application and also helps enforce a consistent set of controls across the enterprise.

3.3 Enterprise Assurance Model

Section 3.2 shows how model based assurance can be used to help automate testing and audit for the individual items within the enterprise architecture. Where assurance is to be given at an enterprise level it is necessary to provide views across slices or the whole of the enterprise architecture.

This section looks at how the assurance modelling approach has been used to provide enterprise architecture risk views by pulling together the information from individual models. The first subsection provides more details of the descriptive elements of the nodes within the assurance model – further subsections then describe how descriptive elements allow models supporting different risk views to be created.

3.3.1 Types and Attributes

The detailed modelling of the controls forms the basic structures from which different views of the enterprise risk are created. As the detailed control model is developed each node within the model is tagged with its type (see Figure 9) within the control framework along with a series of type specific attributes that help describe the entity. For example, if a node represents an application being audited there are attributes naming the application, giving its identity within the enterprise architecture, the business areas and processes it supports along with the criticality of the application. Control objectives have attributes defining identities, risk descriptions and a source where the risk is related to a control objective in a framework such as COBIT.

The framework allows types to be sub-classed and attribute data to be retrieved from a database rather then being defined in a specific model. This helps support generic assurance templates (section 3.2.4) where details can be derived from a source such as the Configuration Management Database (CMDB). This allows, for example, a generic assurance model to be developed for an application where attributes describing the application and its role within the enterprise architecture can be read from a database.

There is a registration process that allows each type to register itself along with interesting attributes including a name and an identity. This provides a way of cataloguing the set of models used to test the individual entities within the assurance system. In this catalogue, one can find, for example, models used on given components in the enterprise directory, models implementing particular elements within the control framework, and manually reported results.

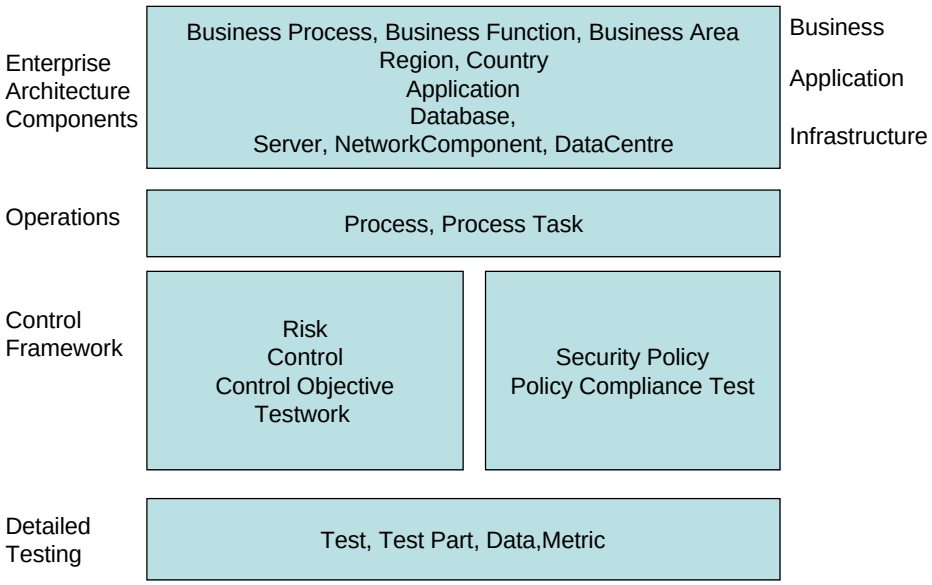

Figure 9:
Types of nodes within the assurance model and
where they fit within the enterprise architecture and risk lifecycle.

3.3.2 Assurance Views for the enterprise architecture

We now have a set of assurance models testing a number of the individual components within the enterprise architecture. Composite models could be built which include servers and databases within the application models and by extension these models would be within assurance models in the context of business processes. However, our experience suggests that it is better to keep each of the base models as simple modular models each just checking a single entity. Composite models are then built on top of these base models where each composite model provides a way of combining the results to provide a view suitable for a particular stakeholder within the enterprise architecture.

For example, rather than the results of server and database compliance testing being included in an application compliance model, a separate application owners view would be created combining the applications that they are responsible for and including references to servers, databases and data centres on which they rely. Creating models in this way allows different priorities to be placed on automated test results for different stakeholders. For example, the business owner will care greatly about the integrity of the information in the database whilst the application owner will be mainly interested in ensuring they manage their risks and run their processes correctly.

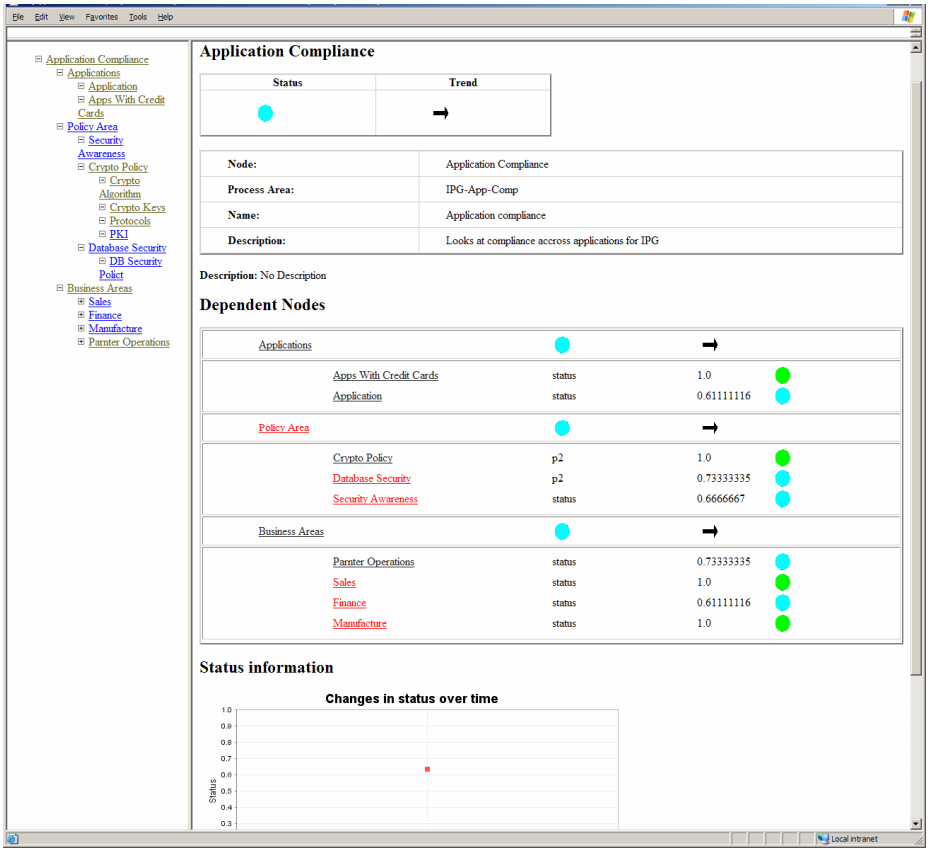

Figure 10:
Example report looking at compliance across multiple applications

Figure 10 shows an example of the first page of a combined report for an organisation's security officer. Here we link to the results of application testing and summarise combined results under different categories that can be defined by the type within the assurance framework and attribute values. Here we can look at all applications but also have a special category of those involved in credit card transactions which may be a particularly high risk category. The "Policy Area" section of the report shows applications according to how well they comply with different security policies. A final grouping shows the applications categorised by different business areas (e.g. finance, sales etc). Providing these different categorisations allows a security officer a quick view of the security of their systems and which parts of the business are affected by issues. The security officers can then dig down through the details to see how each application is performing.

Different stakeholders such as the business owners, director of audit, CFO, CIO and other executives need other views each showing the compliance status according to their concerns. Such views can easily be constructed and customised as concerns change. The hierarchical nature of the model ensures that a high-level dashboard can be created and where there are red lights an executive could dig down into the detail and find out why (or have a member of their staff investigate).

3.3.3 Benchmarking

Benchmarking compliance across an organisation enhances an understanding of where risks may occur and therefore where attention should be focused. Having a set of low-level compliance models based on a common schema behind the companies' control framework allows us to compare across entities within the enterprise architecture. The comparisons can be shown graphically, making it easy for stakeholders to spot systems with results or metrics outside of the norm, and so requiring attention.

Having a consistent way of identifying controls and objectives allows us to perform a much deeper benchmarking. For example, imagine a company wanting to know how well they are performing at disaster recovery across all their systems – normally internal audit would need to commission a separate audit to create such a view. Using the assurance modelling system allows us to search for those control objectives related to disaster recovery over all application models and produce a comparison of the results. This comparison can clearly illustrate the differences between the applications based on the overall status for the control objective and it can include a detailed comparison or list of issues based on the automated testwork for each application. It is the use of a model structure behind the automated analysis that allows this type of benchmarking to be easily addressed and issues spanning the enterprise to be explored.

4 Discussion: Assurance and Risk Management

Section 3 described in some detail the assurance modelling approach, and how it can improve reporting and other aspects of assurance management. In this section we discuss the wider implications of the assurance modelling approach, specifically covering both the wider risk management lifecycle described in section 2, and the emerging attention being paid to shared service-centric enterprise computing.

4.1 Implications for Enterprise Risk Management

4.1.1 A model based repository for risk management

The assurance models described in section 3 largely focused on helping with testing, auditing and reporting of controls as part of the risk life-cycle. However, the assurance model provides a way of capturing the risk and control architecture in a more structured manner than through documents and spreadsheets, and hence, can be used in support of other aspects of the risk life-cycle.

The risk life-cycle has a number of participants and offers a single repository of risks, controls and policies and how they interact. This provides a considerable benefit in helping the participants communicate. The assurance model allows information relating to risks and associated mitigation strategies to be captured and related to elements within the enterprise architecture. Documentation can be created from this information, and by capturing the relevant stakeholders in the enterprise architecture we can communicate this information to them.

The risk information is of course not static and changes to it should lead to changes in the assurance models. These changes will now be reflected in the documentation and the interested parties can be informed.

There are other tools and research that complement our approach and support other aspects of the risk life cycle. CORAS [10] helps model the risks and vulnerabilities during the risk analysis process and Secure Tropos [11] provides a way of modelling the relationships and responsibilities between the various actors within the environment.

4.1.2 Risk analysis and review

Usually companies will have a controls framework for the enterprise architecture. This may provide documentation to understand what risks are being mitigated and when reviews occur. Frameworks such as COBIT provide advice and structure on the types of risks that should be managed in a well run company. These frameworks are paper based documents and transferring these to a structured model allows an enterprise using the assurance modelling to gain an understanding of how their identified risks and mitigations relate to the standards. More importantly as the frameworks evolve a structural comparison between the ideal and the

actual models can be run and used as a basis for introducing new best practice mitigations into the company.

4.1.3 Risk Metrics

The inclusion of risk metrics into the assurance model provides a complementary view on risk to that looking at how well controls (which are intended to mitigate the risks) are being run. Such metrics provide independent verification of the effectiveness of the controls architecture and are indicative that risk has been managed (or has not occurred) for given processes.

For example, a metric monitoring the number of parts inventory held at a factory is indicative of a well run supply chain process. Such a metric may be significant if it gets too small to keep a factory operating or too large indicating too much money is being spent on holding inventory. A further measure of the metric is its stability; too much fluctuation may indicate lack of control even though the extremes are only touched occasionally. Comparing the metric to the results of the controls testing indicates the effect that the controls or lack of them are having on the business. For example, where controls are well run but the metric is showing risk this may indicate that the controls are not effective, and hence, should be reviewed. Alternatively, the metric may show little risk where there are issues with the controls which may be indicative of luck or too many (costly) controls.

4.1.4 Control Architecture Design and Implementation

The assurance model contains details of the controls, policy and audit testwork applied to the various components of the enterprise architecture. As such, it provides a mechanism for understanding the current state of the controls and indicates which items have which controls applied to them. This means that the assurance model can be used as a basis for developing a change plan as new controls need to be introduced from the risk analysis and review process.

It also produces a mechanism for ensuring that as controls are introduced, implemented and subsequently maintained automated testing methods will be in place to go along with the control framework. As the models are changed the automated testing can also be updated so that reports fit the model based control documentation.

4.2 Implications for Shared, Service-Centric Computing

The assurance modelling approach described in this paper is targeted at a single enterprise aiming to gain assurance over the way they run their IT and enterprise architecture. However, there is a periodic trend towards outsourcing where managing assurance and audit programmes can grow in complexity across company boundaries. Some of these issues are addressed in [12] where the assurance models can be used as mechanisms for defining and reporting on the necessary control infrastructure.

Future visions of enterprising computing include more flexible infrastructure [13] and applications formed using a service centric computing vision [14]. From an infrastructure perspective the assurance model can be used to demonstrate the necessary control over shared infrastructures and demonstrate the necessary separations. Research into web service security often concentrates on securing communication protocols and access control rights [15]; however, for enterprises to use services they must understand their operational characteristics [16]. The assurance model provides a structured way of sharing information about the controls implemented by a service and the way that they are managing them [17] hence making the break up of enterprise applications into services achievable.

The assurance system pulls a considerable amount of data from the enterprise and provides a view on risk management on which executives rely. As such, there will be considerable concern about both the integrity and confidentiality of the data. Our current approach is to assume that data is contained within a well managed database, and thus, will have integrity and will be kept confidential. There are cryptographic approaches to chain together audit

messages [18][19][20] that could be used to provide evidence that the data has not been tampered with since entering the audit systems. Trusted computing schemes such as TCG and Database encryption techniques can be used to secure the data; however, there currently does not seem to be a demand for these levels of security.

5 Conclusion

This paper has described an approach of modelling the control architecture of an enterprise and demonstrated how this can be used to automate compliance and audit testing as well as providing different views on enterprise risk for the different stakeholders within the enterprise architecture. Early results from pilots have demonstrated the value of the approach both in terms of helping reduce the workload around audit and the accuracy of the results. We have also demonstrated the value of modelling the controls environment both in providing different views on risk and in support of other aspects of the enterprise risk lifecycle.

References

[1] Spewak, S.H., S.C. Hill. Enterprise architecture planning: developing a blueprint for data, applications and technology. QED Information Sciences (1993).

[2] Armour F.J., Stephen H. Kaisler, Simon Y. Liu, "Building an Enterprise Architecture Step by Step," IT Professional, vol. 01, no. 4, pp. 31-39, Jul/Aug, 1999.

[3] Jonkers, H. van Burren, R. Arbab, F. de Boer, F. Bonsangue, M. Bosma, H. ter Doest, H. Groenewegen, L. Scholten, J.G. Hoppenbrouwers, S. Iacob, M.-E. Janssen, W. Lankhorst, M. van Leeuwen, D. Proper, E. Stam, A. van der Torre, L. van Zanten, G.V. Enterprise Distributed Object Computing Conference, 2003. Proceedings. Seventh IEEE International.

[4] Broder, J.F. *Risk Analysis and the Security Survey*. Elsevier (1999)

[5] ITGI, Control Objectives for Information and Related Technologies (COBIT), 3rd edition, 1998

[6] Sarbanes Oxley Act, see www.sarbanes-oxley.com

[7] Lloyd, V. *Planning to implement service management (IT Infrastructure Library)* .The Stationery Office Books http://www.itil.co.uk/publications.htm

[8] HP, *The HP IT Service Management (ITSM) Reference Model*, 2003

[9] Continuous Control Monitoring: Enabling rapid response to control breakdowns, in research findings of Audit Director Roundtable 2004, http://www.audit.executiveboard.com/ADR/

[10] Vraalsen, F., den Braber, F, Soldal Lund, M, Stolen, K. "The CORAS Tool for Security Risk Management. ". In Proceedings 3rd International Conference on Trust Management 2005 IETF Vol 3477

[11]Giorgini, P. Massacci, F, Mylopoulos, J. and Zannone, N. "Requirements Engineering Meets Trust Management: Model, Methodology and Reasoning. ". In Proceedings 2nd International Conference on Trust Management 2004 IETF Vol 2995

[12] Baldwin, A., Y. Beres, S. Shiu, P. Kearney, *A Model Based Approach to Trust, Security and Assurance.* BT Technical Journal (2006).

[13] Kallahalla, M, M. Uysal, R. Sqaminathan, D.E. Lowell, M. Wray, T. Christian, N. Edwards, C.I. Dalton and F. Gittler. SoftUDC a software based data centre fpr utility computing. Computer November 2004 pp 38-46.

[14] NESSI Strategic Research Agenda, Vol 1 Framing the Future of the Service Centric Economy, Public Draft, available from http://www.nessi-europe.com/

[15] OASIS, Web Services Security (WSS) (http://www.oasis-open.org)

[16] Baldwin, A., Shiu, S., Casassa Mont, M.: Trust Services: A Framework for Service based Solutions, In proceedings of the 26th IEEE COMPSAC, 2002

[17] Baldwin, A., Y. Beres, S. Shiu, Assurance management in service centric computing. HP Labs Technical report (2006)

[18] Baldwin, A. Shiu, S. (2005) Enabling Shared Audit Data, In International Journal of Information Security Volume 4.

[19] Schneier, B., Kelsey, J., "Cryptographic Support for Secure Logs on Untrusted Machines," 7th USENIX Security Symposium Proceedings, USENIX Press, 1998.

[20] Murison, N. and A. Baldwin, Secure Distributed audit for shared customer environments. HP Technical Report 2006