An Efficient Implementation of a Rule-based Adaptive Web Information System

Davide Valeriano, Roberto De Virgilio,
Riccardo Torlone, and Daniele Di Federico

Dipartimento di Informatica e Automazione
Università degli studi Roma Tre
{valeriano,devirgilio,torlone,difederico}@dia.uniroma3.it

Abstract. Mobile devices provide a variety of ways to access information resources available on the Web and a high level of adaptability to different aspects (e.g., device capabilities, network QoS, user preferences, location, an so on) is strongly required in this scenario. In this paper we propose a technique for the efficient adaptation of Web Information Systems. The approach relies on special *rules* that allows us to specify, in a declarative way, how to generate, in an automatic way, the adaptation specifications that satisfy the requirements of a given context. Rules are classified in a set of clusters and a matching relation between clusters and context requirements is defined. The technique of rule evaluation and clustering guarantee that different contexts and orthogonal requirements of adaptation, possibly not fixed in advance, can be efficiently organized and taken into account in the adaptation process.

1 Introduction

Modern Web Information Systems (WIS) are challenged to generate (automatically) a hypermedia presentation built from information that is gathered from different, possibly heterogeneous, sources that are distributed over the Web. Nowadays, given the spread of mobile devices, a novel and fundamental requirement of these systems is the ability to adapt and personalize content delivery according to the *context* of the client. An important point that needs to be taken into account is that this scenario is very dynamic: some aspects of the context can dynamically change. For instance new users with new preferences can be enabled to access the information system, possibly unpredicted types of access devices can be used, alternative network connections can be made available, further aspects of the context (like the location or others environmental factors) need to be incorporated. Also, the technology used to implement the adaptation can be changed and this should have a limited impact on the overall application. It follows that the adaptation process should guarantee a high level of flexibility in terms of responsiveness to rapidly changing requirements of adaptation and suitable actions to be undertaken to meet these requirements.

In the literature, some adaptation approaches rely on the definition of rules to generate suitable adaptation specifications [2, 3, 10]. In [5] we have presented a general methodology for context adaptation, based on special *rules* that model the adaptation at an abstract level, inspired by the approaches cited above. The rules specify, in a declarative way, how to generate, in an automatic way, the adaptation specifications that satisfy the requirements of a given context. A relevant problem is to rapidly select and activate the rules that best fit the requirements of adaptation of the context (in particular when the context changes frequently). In this paper we present an efficient implementation to optimize the activation process of adaptation rules. To this aim, we introduce a clustering technique to classify the adaptation specifications.

The rest of the paper is organized as follows. Section 2 introduces the rule-based approach. In Section 3, we illustrate how to define the set of clusters and the process to efficiently activate a rule, using the clusters. In Section 4 we present a practical implementation and experimental results and finally, in Section 5 we draw some conclusions and sketch future works.

2 A Rule-based approach to model adaptation requirements

In this section we present special rules to specify, in a declarative way, how to generate, in an automatic way, adaptation specifications. The rules are based on two basic notions: the generic profile and the abstract configuration.

2.1 Generic profiles

A (*generic*) *profile* is a description of an autonomous aspect of the context in which the Web site is accessed (and that should influence the presentation of its contents). Examples of profiles are the user, the device, the location, and so on. A *dimension* is property that characterizes a profile. Each dimension can be conveniently described by means of a set of *attributes*. Attributes can be *simple* or *composite*. A *simple attribute* has a value associated with it, whereas a *complex attribute* has a set of (simple or complex) attributes associated with it. In this model, a context is a collection of profiles. Figure 1 reports a graphical example of profiles. In our model, different profiles over the same dimensions can

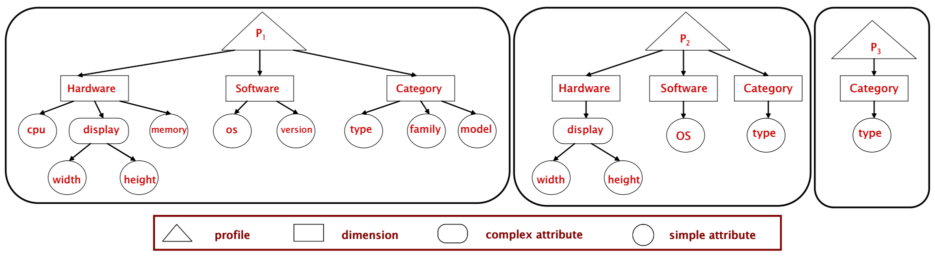


Fig. 1. An example of profiles

be compared making use of a subsumption relationship $\lhd$. Intuitively, given two profiles P_1 and P_2, if $P_1 \lhd P_2$ then P_2 is more detailed than P_1 since it includes the attributes of P_2 at the same or at coarser level of detail. More precisely, we first say that an attribute A *covers* another attribute B if either they are simple and $A = B$ or they are composite and for each sub-attribute A_i of A there is a sub-attribute B_j of B such that A_i covers B_j . The subsumption relationship is then defined as follows.

Definition 1 (Subsumption) *Given two profiles P_1 and P_2, we say that P_1 is subsumed by P_2, $P_1 \lhd P_2$, if for each dimension d of P_1 there is a dimension d' of P_2 such that for each attribute A of d there is an attribute A' of d' covered by A.*

As an example, given the profiles reported in Figure 1 we have that $P_2 \lhd P_1$ and $P_3 \lhd P_2$.

2.2 Abstract configurations

We introduce the notion of (abstract) configuration as a triple $C = (q,\ h,\ s)$ where:

- q is a query over the underlying database expressed in relational calculus, a declarative and abstract language for relational databases ([1]);
- h is an abstract hypertext definition expressed in WebML ([4]), a conceptual model for Web application which allows us to describe the organization of Web pages in a tight and concise way, by abstracting their logical features;
- s is presentation specification expressed in terms of an original notion of logical style sheet, which is a set of tuples over a predefined collection of Web Object Types (WOTs) like text, image, video, form and so on; each WOT τ is associated with a set of presentation attributes: they identify possible styles (e.g. font, color, spacing, position) that can be specified for τ.

An example of configuration is reported below.

$$\begin{aligned} q = \ & \{T: x_1, S: x_2, D: x_3, C: x_4, N: x_5 \mid \\ & \mathit{NewsItems}(N: x_0, T: x_1, S: x_2, D: x_3, G: x_6, J: x_7), \\ & \mathit{Details}(NK: x_0, P: x_8, C: x_4), \mathit{Newspapers}(J: x_7, N: x_5, C: x_9), x_6 = \mathit{Sport}\} \end{aligned}$$

```
h = IndexUnit NewsIndex
      ( source News Items; attributes Title, Date; orderby Date )
    DataUnit ContentData
      ( selector Item = CurrentItem; attributes Title, Date, Content)
    link NewsToDetails
      ( from NewsIndex to ContentData
        parameters CurrentItem = NK)
    link ContentToRestOfContent
      ( from ContentData to ContentData
        parameters CurrentItem = NK)
s = Text( Font: Arial, ..., Border: 0pt)
    Link( Note: False, ..., Color: Blue)
    Image( Format: jpeg, ..., Alignment: left)
```

It is important to note that a configuration is indeed a logical notion that can be represented and implemented in several ways and with different syntaxes. This property guarantees the generality of the approach with respect to actual languages and tools used to implement the adaptive application. For instance, we can implement a configuration using SQL at the content level, XHTML at the navigation level and a set of CSS files at the presentation level.

2.3 Adaptation Rule

The matching relationship between configurations and profiles is represented by means of a novel notion of *adaptation rule*. An adaptation rule has the form $P_r : C_d \Rightarrow C_f$ where:

- P_r is a parametric profile, that is, a profile in which parameters can appear in place of values,
- C_d is a condition, made out of a conjunction of atoms of the form $A\theta B$ or $A\theta c$, where A and B are parameters occurring in P_r, c is a constant value, and θ is a comparison operator,
- C_f is a parametric configuration in which parameters occurring in P_r can appear in place of values.

The intuitive semantics of an adaptation rule is the following: if the client has a profile P_r and the condition C_d is verified then generate the configuration C_f.

An example of adaptation rule for a device profile with the hardware dimension H is the following:

$$\begin{array}{l} H(\mathit{ImgCapable} : X, \mathit{ScreenSize} : Y, \mathit{ColorCapable} : Z) : \\ X = \mathit{false}, Y < \mathit{1500} \Rightarrow \{q, h, s\} \end{array}$$

where X, Y, and Z are the parameters of H and $\{q, h, s\}$ is the following parametric configuration:

$$\begin{array}{rl} q = & \{\, T : x_1, S : x_2, D : x_3, C : x_4 \mid \\ & \mathit{NewsItems}(N : x_0, T : x_1, S : x_2, D : x_3, G : x_5, \\ & J : x_6), \mathit{Details}(\mathit{NK} : x_0, P : x_7, C : x_4)\} \end{array}$$

```
h =  Page NewsPage ( units NewsIndex )
     Page ContentPage ( units ContentData )
     IndexUnit NewsIndex
       ( source News Items; attributes Title, Date;
         orderby Date )
     DataUnit ContentData
       ( selector Item = CurrentItem;
         attributes Title, Date, Content)
     link NewsToDetails
       ( from NewsIndex to ContentData
         parameters CurrentItem = NK)
s =  Text( Color: Black, Size: 8pt)
     Link( Style: Underline, Color: Black)
     Image( Size: Y × 0,5, Color: Z)
```

Note the use of the parameters Y and Z of H to resize the images and to eliminate/maintain colors.

A profile P *activates* a rule $P_r : C_d \rightarrow C_f$ if $P_r \lhd P$. Hence, a profile P can activate a rule for a profile that is more general than P. Now, let R be a rule $P_r : C_d \rightarrow C_f$ activated by a profile P and let C be the configuration obtained from C_f by substituting the parameters of P_r with the actual values occurring in P: we say that C *is generated* by P using R.

3 An optimization strategy

We assume to have at disposal an initial set of rules $\mathbf{S_R}$. In an adaptability scenario, context changes generate events that produce one or more profiles (instances). Each profile (instance) can activate a rule of $\mathbf{S_R}$. We should optimize the research to select and activate efficiently the most appropriate rules and generate the most suitable configuration. So we organize S_R classifying the rules in a set of clusters. Each cluster represents a *class* of adaptation that matches the requirements of a *class* of context.

3.1 A distance between profiles

We define a distance that allows to compare profiles. We take advantage of the tree structure of the generic profile. We get inspiration from the *Tree Edit Distance* [9], where the distance between two trees T_1 and T_2, that we indicate as $d_{ted}(T_1, T_2)$, depend on the sequence of operations (chosen in a limited predefined set) that *transforms* T_1 into T_2. We assign a fixed cost to each operation: the distance between the considered trees is the sum of essential costs for the transformation. For our case, let's consider the set of operations *edit*, *remove* and *add*, respectively to modify the content of a node, to delete a node and to create a new node. In our tree structures, we distinguish two principal types of node: dimension and attribute. We have operations to modify, remove or create dimensions and attributes and we indicate the cost of an operation with γ.

Definition 2 (S-derivation) *Given two trees T_1 and T_2, $S = s_1, \ldots, s_k$ the sequence of operations to transform T_1 in T_2, an S-derivation between T_1 and T_2 is a sequence $U_0, \ldots, U_k$ of trees resulting by applying the single operation, where $U_0 = T_1$ and $U_k = T_2$, and the cost $\gamma(S - derivation)$ is $\sum_{i=0}^{i=k} \gamma(s_i)$.*

So we can intuitively define the distance between two profiles P_1 and P_2, the minimum cost for a S-derivation between them.

Definition 3 (Distance between profiles) *Given two profiles P_1 and P_2, we define the distance between them as $d_{ted}(P_1, P_2) = min\{\gamma(S - derivation) \mid$ S-derivation from P_1 to $P_2\}$*

3.2 Definition of Clusters

Let's represent a cluster as a tree where the nodes are profiles. The *root* represents the minimum set of information that nodes of the tree share. Basically a client is identified by the profile sent to the system. Let's remember that a profile is principally described by dimensions, that characterize the main information, articulated in attributes. So, the root of a cluster is a profile characterized only by dimensions. It will contain the minimum information that several profiles has to include to be in the same cluster. Given a profile P, pruned by all attributes, we can build several beginner clusters, rooted with profiles generated from P, on the number and type of dimensions. However the designer can decide the detail level for the roots. For instance a designer can decide to articulate some dimensions of a root with attributes, to extend the level of detail of the minimum information to share.

We define a cluster as follows

Definition 4 (Cluster Tree) *A cluster CL is a couple* $\{N_{CL}, E_{CL}\}$ *where* N_{CL} *is the set of nodes (profiles) and* E_{CL} *is the set of edges* (n_i, n_j) *such that* $n_i \lhd n_j$

We can relate a node P_r of a cluster with the rule $P_r : C_d \rightarrow C_f$ in an Hash table using P_r as access key.

Given the set of roots $R_t = \{R_{t_1}, \ldots, R_{t_n}\}$ and the set of rules S_R, we initialize a set of clusters $CL = \{CL_1, \ldots, CL_n\}$ as follows:

- we set $CL_i = \{\{R_{t_i}\}, \emptyset\}$,
- $\forall$ $(P_r : C_d \rightarrow C_f) \in S_R$, (i) we search a CL_i such that $R_{t_i} \lhd P_r$, and we navigate the cluster in depth until level k where doesn't exist any node subsumed by P_r, (ii) we search at level $k-1$ the node Nd such that $Nd \lhd P_r$ and if there are more nodes Nd such that $Nd \lhd P_r$ we choose the node with lowest distance, (iii) we add P_r to N_{CL_i} and (Nd, P_r) to E_{CL_i}, (iv) if there are one or more nodes Nd_i such that $P_r \lhd Nd_i$ then we delete any edge of type (Nd_j, Nd_i) in E_{CL_i} and add to it the edge (P_r, Nd_i),
- for each node of a cluster we add the relative rule to the Hash table.

3.3 The activation process

Built the set of clusters, now we can perform an optimization strategy to activate a rule in an efficient way. Given the set of rules S_R, the set of clusters $CL = \{CL_1, CL_2, ..., CL_n\}$, the Hash table H_t and a profile instance P_I of a profile schema P_S, we produce a set of configurations as follows:

1. we initialize P' with P_S;
2. we search the cluster CL_i in CL, with a root R_{t_i} such that $R_{t_i} \lhd P'$ and that has the lowest distance from it;
3. (i) we navigate in depth the cluster until level k where any node, that is subsumed by P', doesn't exist; (ii) we search at level $k-1$ the node Nd such that $Nd \lhd P'$ and if more nodes are subsumed by P' we choose the one with

lowest distance; (iii) we extract the rule R_l, related in the H_t to Nd, and if the condition is satisfied, we activate R_l and instance the configuration with the value of P_I; (iv) we update P' as P' - Nd;

4. we iterate from step (2.) until P' is not empty and a rule that can be activated by P' exists;
5. if no rule can be activated in this way, the system will return a default configuration.

3.4 An example of interaction between profiles and clusters

Let's assume a client sending a profile P_I (see Figure. 2(a)), instance of a profile schema P to the system. Assume also that the system has already initialized its own set of clusters, composed by clusters C_1 and C_2 (see Figure 3). We easily see that the root of the cluster C_2 is the one which subsumes P and has the lowest distance from it. So we navigate that cluster and note that the only child N of the root it is equal to P; then we extract the rule R_l related in the H_t to N and if the condition is satisfied, we activate it and instance the configuration with the values of P_I. Subsequently, assume that a new profile P'_I (see Figure 2(b)), instance of a profile P', is sent by another client to the system. At the same way explained before, we see that the root of the cluster that subsumes P' and has the lowest distance from it, is cluster C_1 root. Navigating that cluster, we need to confront P' with root's two children. We easily see that no one of them is equal to or has a subsumption relation with P': being their parent a root, is not possible to activate its rule; so, as we explained before, the system will return a default configuration.

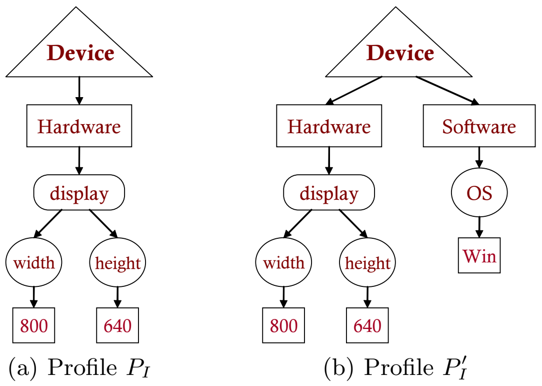
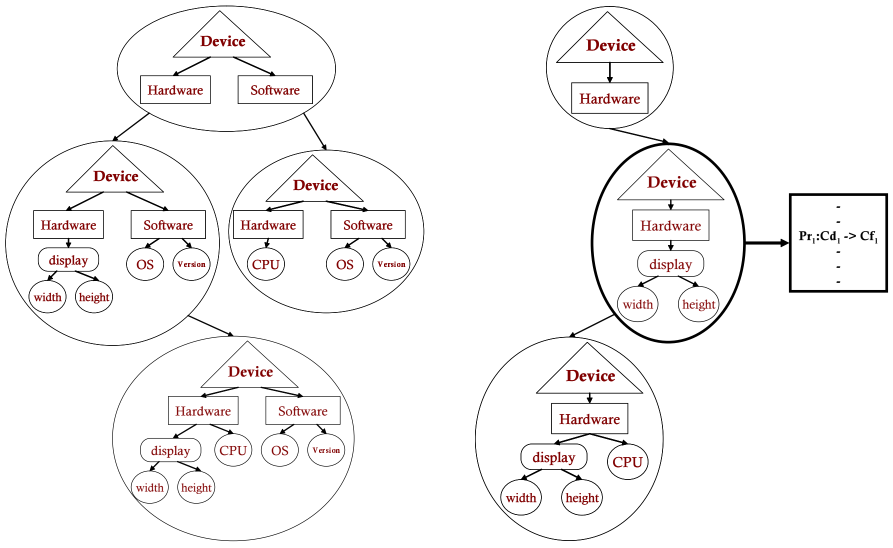


Fig. 2. Profiles sent to the system.



(a) Cluster C_1 (b) Cluster C_2

Fig. 3. Cluster Set

4 A practical implementation

4.1 Benefits of Clustering

We can easily see that using this clustering algorithm we can reduce the number of accesses to S_R. In our system, we have seen that a profile given by a client can activate a rule in the default set. So when a client sends a profile to the system it has to search for the most suitable rule in S_R. We can assume that the worst case is when the system has the greatest possible number of default rules, that is when the system has in S_R one rule for each possible profile that a client can send. Without using clusters we should execute a number of ordered accesses equal to the total number of possible profiles. It is easy to see that the number of possible profiles is the number of profiles obtained with every possible different combination of dimensions and attributes. Assuming to have a complete schema with n dimensions and m attributes for each dimension, we can see that the number of different combinations of dimensions is $\sum_{i=1}^{n} \binom{n}{i}$ while the number of different combinations of attributes for each single dimension is $\sum_{j=1}^{m} \binom{m}{j}$. Combining these two results and looking carefully at our structure, we see that the number of different profiles is equal to $\sum_{i=1}^{n} \left[\sum_{j=1}^{m} \binom{m}{j}\right]^{\binom{n}{i-1}} (n-i+1)$. If we use the clustering algorithm, in this case we will obtain a number of clusters equal to the number of different combinations of dimensions, seen above. To access the right clusters, the system has to access to all of their roots and then find the right way on the selected tree. So it can ignore a number of trees equal to $\sum_{i=1}^{n} \left[\binom{n}{i}\right] + 1$. We can see that in this situation the worst case is when the root of the selected cluster has the maximum number n of dimensions, because it would be the tree with the maximum number of nodes. The remaining clusters will contain a number of profiles equal to $\sum_{i=1}^{n-1}\left[\sum_{j=1}^{m}\binom{m}{j}\right]^{\binom{n}{i-1}}(n-i+1)$. These profiles will be ignored by the system, which has only accessed the roots of their clusters, that are not profiles. So, counting also these accesses, we find that *at least* we obtain to save a number of accesses equal to $\sum_{i=1}^{n-1}\left[\sum_{j=1}^{m}\binom{m}{j}\right]^{\binom{n}{i-1}}(n-i+1)-\sum_{i=1}^{n}\binom{n}{i}$. This number doesn't take count of the fact that the system won't access all the nodes of the selected cluster, but only some of them, depending on its structure. Anyway it gives evidence that, using this clustering algorithm, it is possible to obtain a certain benefit in terms of number of accesses.

4.2 Experimental results

We have designed and developed a practical implementation to test the our approach. The system relies on an RDF database of contents and the selection of data is done by performing RDF queries expressed in SPARQL [8]. We use an RDF(S) representation for schema level and RDF for instance level. The implementation makes use of Jena [6], a Java framework for building RDF based applications. Each rule is implemented in terms of Inference Rule of Jena [7]. Using the Inference Engine of Jena, the system generates a configuration that infers the adaptation on RDF(S) documents. Finally the system returns several response as output, using XSLT transformations on RDF instances. We have tested our system to experiment the effectiveness and the efficiency of the approach illustrated in this paper in several practical cases. On the server side, we used an IBM computer xSeries 225, equipped with a Xeon2 2.8Ghz processor, a 4 GB RAM, and a 120 GB HDD SCSI. The Web site have been accessed by three different types of devices: a mid-range desktop, several PDAs with different capabilities, and some cellular phones. Figure 4 reports the average response time (vertical axis) for each device type (horizontal axis). The Figure shows promising times to produce the adapted response. The Desktop client presents an average time of 72 ms, the PDA client 97 ms and the Wap client 121 ms. The diagram compares the medium response time in absence and in presence of clusters. In particular the system, using clusters, capitalizes the approach for mobile devices, almost halving the response time. For instance the Wap client benefits from 121 ms to 59 ms.

5 Conclusions and future works

In this paper we have presented a novel rule-based approach supporting the automatic adaptation of content delivery in Web Information Systems. This problem arises in common scenarios where the information system is accessed, in different contexts, by a variety of mobile devices. The adaptation is achieved automatically by means of special rules that specify, in a declarative way, how a configuration can be generated to meet the requirements of adaptation of a

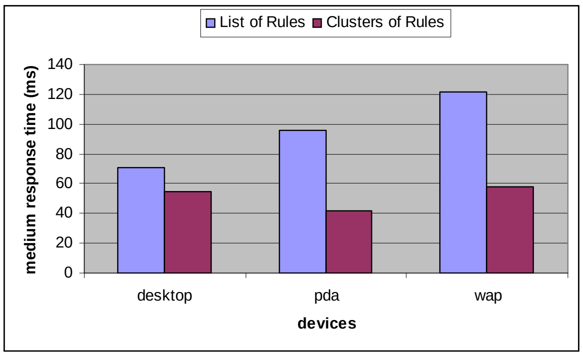


Fig. 4. Results of experiments

given profile. Finally the organization of rules in a set of clusters guarantees the responsiveness of the system in a dynamic scenario. Different contexts and orthogonal requirements of adaptation, possibly not fixed in advance, can be taken into account in this process. The results presented in this paper are subject of further conceptual and practical investigation. From a conceptual point of view, we are currently investigating the expressive power of rules and the generality of clusters. In particular we are improving the distance between profiles, using a comparison between common pathes. From a practical point of view we are improving the efficiency of the system and we are extending its functionality.

References

1. S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases.* Addison-Wesley, 1995.
2. C. Bettini, D. Maggiorini, and D. Riboni. Distributed context monitoring for continuous mobile services. In *IFIP TC8 Working Conference on Mobile Information Systems (MOBIS)*, 2005.
3. S. Ceri, F. Daniel, V. Demaldé, and F. M. Facca. An approach to user-behavior-aware web applications. In *5th International Conference on Web Engineering (ICWE)*, pages 417–428, 2005.
4. S. Ceri, P. Fraternali, A. Bongio, M. Brambilla, S. Comai, and M. Matera. *Designing Data-Intensive Web Applications.* Morgan Kaufmann Publishers Inc., 2002.
5. R. DeVirgilio, R. Torlone, and G.-J. Houben. A rule-based approach to content delivery adaptation in web information systems. In *7th International Conference on Mobile Data Management (MDM'06) - to appear -*, 2006.
6. HP-Labs. Jena. http://jena.sourceforge.net/, 2004.
7. HP-Labs. Jena 2 inference support. http://jena.sourceforge.net/inference/, 2005.
8. Hp-Labs. Sparql, query language for rdf. www.w3.org/TR/rdf-sparql-query/, 2005.
9. C. Isert. The editing distance between trees. Technical report, Ferienakademie, for course 2: Bume: Algorithmik Und Kombinatorik, 1999.
10. H. Kiyomitsu, A. Takeuchi, and K. Tanaka. Activeweb: Xml-based active rules for web view derivations and access control. In *International workshop on Information technology for virtual enterprises (ITVE01)*, pages 31–39, 2001.