
Techniques for mining the design of e-government services to enhance end-user experience

Yiannis Panagis, Evangelos Sakkopoulos*,
Athanasios Tsakalidis and Giannis Tzimas

Computer Engineering and Informatics Department,
University of Patras,
GR-26504, Rio Patras, Hellas

Research Academic Computer Technology Institute,
61 Riga Feraiou Street, GR-26221, Patras, Hellas
E-mail: panagis@ceid.upatras.gr
E-mail: sakkopul@ceid.upatras.gr
E-mail: tsak@cti.gr E-mail: tzimas@cti.gr

Spiros Sirmakessis

Technological Educational Institution of Messolongi,
Department of Applied Informatics
in Administration and Economics,
GR-30200, Messolongi, Hellas

Research Academic Computer Technology Institute,
61 Riga Feraiou Street, 26221 Patras, Hellas
E-mail: syrma@cti.gr

*Corresponding author

Miltiadis D. Lytras

Research Academic Computer Technology Institute,
Argolidos 40-42, 153-44 Gerakas, Attica, Greece
E-mail: lytras@ceid.upatras.gr

Abstract: The World Wide Web solutions are increasingly endorsed by several levels of government in order to deliver all kinds of the so-called e-government services. To face the complexity of joint design and development procedures, it is imperative to adopt a web modelling language such as the WebML. In this work, we propose a methodology that enhances the ability to detect, during the early development stages, similar design snippets that perform the same functionalities. Additionally, a methodology to infer frequent constructs at the design level, when applied to similar e-government services, can lead to safe conclusions as for when a specific construct constitutes a design pattern. Finally, we illustrate a validation scenario.

Keywords: electronic government; web design patterns; web modelling; knowledge mining; web mining; web engineering.

Reference to this paper should be made as follows: Panagis, Y., Sakkopoulos, E., Tsakalidis, A., Tzimas, G., Sirmakessis, S. and Lytras, M.D. (2008) 'Techniques for mining the design of e-government services to enhance end-user experience', *Int. J. Electronic Democracy*, Vol. 1, No. 1, pp.32–50.

Biographical notes: Yannis Panagis holds a PhD from the Computer Engineering and Informatics Department at the University of Patras and he is a member of the Research Unit 5 of the Research Academic Computer Technology Institute. He holds an MSc from the same Department, where he has also completed his undergraduate studies. His interests span the areas of data structures, string processing algorithms and web engineering, where he has published papers in international journals and conferences.

Evangelos Sakkopoulos holds a PhD from the Computer Engineering and Informatics Department, University of Patras, Greece, and he is a member of the Internet and Multimedia Technologies Research Unit of the Research Academic Computer Technology Institute. He has received the MSc with honours and the Diploma of Computer Engineering and Informatics at the same institution. His research interests include e-government, information systems, web engineering, web services, web usage mining, web-based education, web searching and intranets. He has more than 40 publications in international journals and conferences at these areas.

Athanasios Tsakalidis is a Professor in the Department of Computer Engineering and Informatics of the University of Patras since 1993. He completed his Diploma in Mathematics in 1973 (University of Thessaloniki), his studies and his PhD in Informatics in 1980 and 1983 respectively (University of Saarland, Germany). From 1983 to 1989, he worked as a researcher at the University of Saarland, and was student and cooperator of Professor Kurt Mehlhorn (Director of Max Planck Institute of Informatics in Germany). His main interests are data structures, graph algorithms, computational geometry, multimedia, information retrieval and bioinformatics.

Giannis Tzimas is the technical coordinator of the internet and Multimedia Technologies Research Unit in Research Academic Computer Technology Institute (<http://www.cti.gr>). He holds a PhD from the Computer Engineering and Informatics Department, University of Patras. He has significant professional experience in web applications' design and development and he was the project manager of several applications such as the SOPHIA intranet of the Athens 2004 Organising Committee. His research interests include web engineering, web modelling, web-based education and intranets/extranets. He has more than 35 publications in international journals and conferences at these areas.

Spiros Sirmakessis is an Associate Professor in the Technological Education Institute of Messolongi and the Manager of the Internet and Multimedia Technologies Research Unit of the Research Academic Computer Technology Institute (<http://www.cti.gr>). He is the coordinator of the NEMIS project (<http://nemis.cti.gr>), a network of excellence in text mining and responsible for the web mining working group. He is the editor of the book *Text Mining and Applications*, published by Springer Verlag (2003) and author of three books and several research papers published in international journals and conferences.

Miltiadis D. Lytras is the Editor-in-Chief of *International Journal of Knowledge and Learning* (<http://www.inderscience.com/ijkl>) as well as the Editor-in-Chief of *International Journal of Teaching and Case Studies*

(<http://www.inderscience.com/ijtcs>). He serves also as the Executive Editor of *International Journal on Semantic Web and Information Systems* (<http://www.idea-group.com/ijswis>). He is a faculty member in the Computer Engineering and Informatics Department-CEID (University of Patras), and in the Department of Business Administration-BMA (University of Patras). His research focuses on semantic web, knowledge management and e-learning, with more than 80 publications in these areas.

1 Introduction

The internet and particularly the web has been warmly endorsed by enterprises and the academic community and it is not surprising to see several levels of government utilising more and more of these approaches to deliver all kinds of state services. While trying to operate more efficiently and upgrade the quality of service, government interest in the new Information Technologies continues to expand. In fact, the development of electronic government environments and services has received great attention from governments all over the world, as it is discussed in the eEurope 2005 (2005) action plan for the European Union (EU), in the recent eEurope 2010 challenges (2005), and in the eGov (2002) for the USA.

Although electronic government exists in such a wide agenda, there is no single and agreed definition for ‘*e-government services*’. However, a definition can be developed based on its various uses. McClure (2000) described electronic government as government’s use of technology, particularly web-based internet applications, to enhance the access to and delivery of government information and service to citizens, business partners, employees, other agencies, and government entities. Golden et al. (2003) proposed that electronic government consists of using technology, particularly the internet, as a means to deliver services to citizens, businesses and other entities with the purpose of providing convenient access to government information and services. In Wang et al. (2005), a definition for ‘*e-government services*’ is understood as the information and services provided to the public on government websites.

Currently, the design trends include creating the so-called ‘*citizen-oriented websites*’ where content and services are organised around the anticipated needs of web visitors. This approach is widespread and it has resulted in commonplace helpful features as website search, links to associated government agencies and personalised interface.

However, despite the attempt for wider penetration of e-government services, little effort has been placed in devising mechanisms and applying techniques to evaluate them. Evaluation is a necessary activity for ensuring return on investment over time. In e-government, the investment comes both from the state who provide the services as well as from the citizens that consume them. In particular, financial investment includes spending on infrastructure and R&D to develop and implement e-government services. In parallel trail, the organisational investment exists, which tends to be unobservable sometimes, and includes the cost in time and man-power that government agencies spend to rethinking and re-engineering the service delivery system of an existing procedure into an e-service. To consider the citizens’ return on investment, it helps to assume first that, at a minimum, the cost for any specific service is less when provided using the web than through the traditional alternatives and as a consequence, each web interaction

represents a cost savings. Even in cases without cost savings for the government, the cost of access by citizens can be the source of benefit, often in reduced travel and work-time cost. As a result, to obtain the discussed benefits, government services need to move themselves and their customers from the traditional service delivery procedures to e-government services.

The main challenge in the e-government solutions is to produce citizen-friendly solutions. A lot of attention has to be given at the variation in the type of services being provided. Governments typically provide services that enterprises do not, in most cases without distinguishing different markets. Therefore, this semantic distinction is fundamental to perceive the difference between the public organisation's behaviour and private firms, and furthermore, to define the distinct means and methods of service providing (Rainey et al., 1976; Rainey, 1979; Rainey and Bozeman, 2000).

The former difference has to do with the 'exclusive' nature of services provided by e-government infrastructure. A lot of e-government services, such as submitting financial VAT or tax reports, are 'exclusively' available, i.e. they can not be performed by any other individual, organisation or enterprise than the government. In this sense, designing web solutions that make customers excited to be in a specific website (von Dran et al., 1999) is in fact a competitive advantage for a commercial site, but makes no difference for an e-government site, which provides payment services for the income tax. The case of e-government is very different from services provided in the private sector, which can be consumed at a competitor's websites as well when not satisfactory. Clearly, the interaction between the public and government agencies over the web is different from the online customer-company interactions.

E-government services have to face one more difference in comparison with the typical e-commerce web, which has to do with the broader heterogeneity in the user base. The design of e-government services has to deal with greater variation in the users' characteristics. As a consequence, the design and the personalisation techniques of an e-government service involve more groups of experts with diverse backgrounds than any other 'specialised' target group's solution such as e-shops or e-learning environments.

To face the complexity of joint design and development procedures, a web modelling language such as the WebML is usually adopted (details of WebML are presented later in Section 3). However, even if such is the case, lots of inconsistencies, debugging overheads and moderate code reusability have a fair chance to appear.

Motivated by the ever-increasing complexity of e-government applications, we propose in this work a methodology that enhances the ability to detect, during the early development stages, similar design snippets or even larger design constructs that perform the same functionalities, to increase implementation consistency, application maintenance and quality of e-government service. Additionally, a methodology to infer frequent constructs at the design level, when applied to similar e-government services, can lead to safe conclusions as for when a specific construct constitutes a design pattern. In the following, we assume the usage of WebML for applying our methodology.

The remainder of this paper is organised as follows: Section 2 presents background work concerning e-government services and discusses modelling methods. Section 3 provides a quick overview of WebML. Section 4 presents in detail a methodological approach for identifying reusable design solutions within the conceptual schema of web applications, while Section 5 illustrates a validation example of the proposed methodology in an instance of an application scenario. Finally, Section 6 provides concluding remarks and discusses future steps.

2 Background and related work

E-government is a vision towards the digital future cultivated since the mid-1990s, which has been obstructed by various challenges. The World Markets Research Centre has performed a detailed study of 2288 national governmental websites in 196 countries and it has found that only 8% of the websites offered services executable online and only 6% provided integrated services at their portals (World Markets Research Centre, 2001). According to America's General Services Administration (2000), a distinguishing solution is 'e-citizen', Singapore's e-government portal. It generates approximately \$14.5 million in savings annually for the Singapore government and was rated second in Accenture's 2001 e-government survey (Accenture Consulting, 2001). In this direction, a lot of e-government services are currently being developed.

Concerning the evaluation of e-government services, limited work has been done. Eschenfelder et al. (1997) has worked in website evaluation for the federal government of USA. Demchak et al. (2000) proposed Website Attribute Evaluation System (WAES), which is designed for evaluating the organisational openness of a government website solely from characteristics of the website itself. Moreover, West (2000) developed an evaluation approach on the basis of characteristics found by observing websites.

In this work, we face the evaluation of an e-government service from a different viewpoint, that of the web design and development cycle. In the web engineering community, several web application modelling methods have been proposed to tackle web design and development setbacks, primarily based on the key principle of separating data management, site structure and page presentation. Hypermedia applications inspired the proposal of Relationship Management Methodology (RMM) (Isakowitz et al., 1995) and HDM (Garzotto et al., 1993), which pioneered the model-driven design of hypermedia applications and influenced several subsequent proposals like HDM-lite (Fraternali and Paolini, 1998), a web-specific version of HDM, Strudel (Fernandez et al., 1998) and OOHDM.

Object-Oriented Hypermedia Design Method (OOHDM) (Schwabe and Rossi, 1998) is concerned with the conceptual modelling, navigation design, interface design, and implementation of hypermedia applications. Navigational contexts in OOHDM provide a rich repertoire of fixed navigation options.

There also exist several proposals for using UML (Booch et al., 1998) for modelling the architecture of web applications. Webpages are modelled as UML components, distinguishing among their 'server side aspects' (i.e., their relationship with middle tiers, databases, and other resources) and their 'client side aspects' (i.e., their relationships with browsers, Java applets, ActiveX controls and so on). Some extensions to the UML notation have been proposed by Conallen (1999a, 1999b).

Finally, Web Modeling Language (WebML) (Ceri et al., 2000) builds on several previous proposals for hypermedia and web design, including HDM, HDM-lite, RMM and OOHDM. It provides graphical, yet formal specifications, incorporated in a complete design process, which can be assisted by visual design tools for expressing a hypertext as a set of pages made up of linked content units and operations, and for binding such content units and operations to the data they refer to. WebML provides a model-driven approach to website development, which is a key factor for defining disciplined development processes for the construction of complex sites, supporting advanced features like multi-device access, personalisation, and evolution management. In this work, we have chosen WebML to work on the proposed methodology, mainly because it

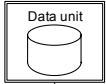
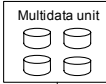
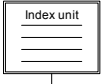
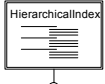
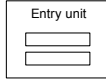
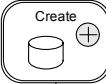
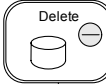
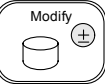
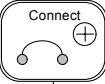
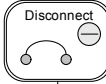
is supported by a powerful CASE tool, WebRatio (2005), which provides efficient automation in the design process.

3 WebML at a glance

WebML is a visual language for specifying the content structure of a web application and the organisation and presentation of contents in one or more hypertexts (Ceri et al., 2002). The first step of designing in WebML is to specify the data schema of the web application, to express the organisation of contents using E-R primitives. The next step is *Hypertext Design*, which produces schemes expressing the composition of content and the invocation of operations within pages, as well as the definition of links between pages.

Site views, areas, pages and content units constitute the overall structure of the hypertext. A *site view* is a hypertext, designed to address a specific set of requirements, according to the different users or the different publishing devices. It is partitioned into *areas*, which in turn may consist of other *sub-areas* or *pages* with a homogeneous purpose. Pages are the actual containers of information delivered to the user; they are made of content units that are elementary pieces of information, extracted from the data sources by means of queries and which are published within pages. In particular, as described in Table 1, *content units* denote alternative ways for displaying one or more entity instances. To specify a data, multidata, index and scroller unit, the designer should define a *source* (the entity from which unit content is extracted) and a *selector* (a condition used for retrieving the actual objects of the source entity that contribute to the unit's content).

Table 1 Some basic WebML elements. The whole set is described in Ceri et al. (2002)

				
Entity [Selector]	Entity [Selector]	Entity [Selector]	Entity1 [Selector1] NEST Entity2 [Selector2]	Entity [Selector]
It displays a set of attributes for a single entity instance	It displays a set of instances for a given entity	It displays a list of properties, also called descriptive keys, of a given set of entity instances	It is a variant of index, in which the index entries are organised in a multi-level tree	It represents forms for collecting input values into fields
				
Entity <param := value>	Entity [conditions]	Entity [Conditions] <param := value>	Relationship	Relationship
It enables the creation of a new entity instance	It deletes one or more objects of a given entity	It updates one or more objects of a given entity	It creates new instances of a relationship	It deletes instances of a relationship

Within site views, content units and pages are interconnected in a variety of configurations by means of *links*, yielding composite navigation mechanisms. Besides representing user navigation, links between units also specify the transfer of certain information (called *context*) that the destination unit uses for selecting the data instances to be displayed.

Apart from content publishing, WebML allows specifying data update operations, like the creation, modification and deletion of instances of an entity, or the creation and deletion of instances of a relationship (Table 1). Operations do not display data and are placed outside pages. Special purpose operations, such as e-mail sending, login, logout, and e-payment, can also be specified (Brambilla et al., 2002).

4 Mining reusable designs: methodology and metrics

4.1 Web design patterns and web application frameworks

The notion of design patterns as tools that describe a piece of design experience or expert advice and make it reusable, was initially conceived by the architect C. Alexander, in the context of architecture and urban planning (Alexander et al., 1997):

“... Each pattern describes a problem which occurs over and over again in our environment, and then describes the core solution to that problem, in such a way that you can use this solution a million of times over ...”

Nowadays, the use of patterns has been further extended in a diversity of domains. In the field of software engineering, design patterns are increasingly used to capture expertise in object-oriented programming (Gamma et al., 1995). In this case, design patterns capture common design problems, study different solutions, document good designs that solve problems faced during any serious software development project and analyse the trade-offs. More recently, design patterns have been introduced in the web modelling field as well, for describing the navigation and structure of web applications (Schwabe et al., 1997, 2001; Bernstein, 1998; Nanard et al., 1998; Garzotto et al., 1999). The availability of design patterns, which offer verified solutions to typical page configuration requirements, further facilitates the task of the hypertext architect and enforces a coherent design style over large and complicated applications, augmenting hypertext regularity and usability.

Taking the use of web design patterns one step further, the notion of web application frameworks for specific domains was introduced (Schwabe et al., 2001; Ceri et al., 2001). Although the use of web design patterns is valuable when building a web application, complex web applications need to maximise reusing larger design structures (the whole navigation topology or structure plus associated functionality). These design structures may arise at the application (conceptual level) or during navigational design. For example, the way users navigate to locate and purchase items online is usually similar regardless of the site. Designers should find ways to express these commonalities by using generic designs such that only aspects unique to a particular site should be designed or programmed. Based on the above, web application frameworks capture the design decisions that are common to a category of websites and provide reusable design for that category (Ceri et al., 2001).

A primitive set of design patterns has already been identified in WebML, comprising compact and consistent, one-step solutions, applicable in real-life scenarios of web applications. Patterns have been discovered for data design, by identifying typical roles of information objects within the data schema (i.e., core concepts, interconnection concepts, access facilitators), and typical data sub-schemas constructed around such roles (i.e., core, interconnection, access, personalisation sub-schema) (Ceri et al., 2004). Patterns have also been defined for *hypertext design*, by identifying unit compositions representing typical hypertext navigation chains and *content publishing* (cascaded index, filtered index, filtered scrolled index, guided tour, indexed guided tour, object viewpoint, nested data, hierarchical index with alternative sub-pages). Moreover, WebML also introduces patterns for *content management* operations (object creation/deletion/modification, relationship creation/deletion, create/connect pattern, cascaded delete) (Ceri et al., 2002).

A *pattern* in WebML typically consists of a *core specification*, representing the invariant WebML unit composition that characterises the pattern, and a number of *pattern variants*, which extend the core specification with all the valid modalities in which the pattern can start (*starting variants*) or terminate (*termination variants*). Starting variants describe which units can be used for passing the context to the core pattern composition. Termination variants, on the other hand, describe how the context generated by the core pattern composition is passed to successive compositions in the hypertext (Fraternali et al., 2002).

4.2 The methodology

In what follows, we present in detail a methodology for mining recurrent design solutions in the conceptual schema of applications modelled using WebML. Our objective is to capture compositions of hypertext elements (pages, units, operations, links) serving several application purposes. Examples could be the arrangement of pages, units, and links for supporting the navigation between a number of core objects, or for accessing a core object via one or more access objects and creating a new object through an operation.

These configurations are captured in the process of (or after) modelling a web application, are complementary to the WebML predefined set of patterns and can serve as effective design solutions enabling reuse, consistency and quality improvement in the development and maintenance process. The methodology can be applied to a large number of web applications, to assist in the identification of templates for web application frameworks of specific domains (i.e. e-government). Moreover, it can be applied for the discovery of new design patterns extending the predefined set of patterns supported by WebML, since the process of finding new patterns involves analysing successful applications and reverse-architecting its underlying design structure (Schwabe et al., 1997). In the remainder of the paper, we will refer to these configurations as ‘candidate patterns’ or ‘design constructs/solutions’.

In the sequel, we present in detail the seven steps of the extraction mechanism.

Steps 1 and 2: Conceptual Schema Initialisation

- Iteratively, we traverse each site view of the web application’s conceptual schema and search for the existence of predefined WebML patterns (content publishing and content management patterns) taking into account their variants. Every pattern found is stored in a *pattern occurrences repository*, along with its starting and termination variants. The repository also stores the occurrence frequency of each pattern. Figure 1 depicts the retrieval of a filtered index, within a site view. The above can be achieved using XSL (Fraternali et al., 2002). The XSL language (XSL, 2001) allows writing pattern-matching rules that can be applied to an XML document for generating a new XML document. Each rule contains a matching part for selecting the target XML elements, and an action part to transform the matched elements. The XSL documents serve therefore the purpose of extracting the instances of patterns from the XML specification of the WebML conceptual schema.
- The purpose of this step is to create a more uniform conceptual schema, thus enabling the easier extraction of design constructs in the steps to follow. Taking into account the various predefined WebML pattern variants, and utilising XSL rules, we substitute, where possible, the variants found within each site view with a default pattern variant. The default pattern variant is the one having the maximum occurrence frequency (e.g., if we find a modify pattern and its termination variant having the larger occurrence frequency is the same page termination variant, we use it as the default pattern and substitute all the other variants). This step requires the designer’s intervention to assure that the substitution does not lead to inconsistencies in the conceptual schema. In general, the substitution is feasible when no context is passed to the starting or termination variant. In case that more than one pattern is assigned the maximum frequency, we choose the first found. Upon completion of this step, we have a more uniform XML definition of each site view.

Step 3: Design Solutions Extraction

- We traverse each area, sub-area and page of all the newly generated site views, to locate identical configurations of hypertext elements along with their variants, either within a site view or among different site views, using the methodology presented in Section 4.3. The configurations retrieved should not already belong to the predefined WebML set of patterns but may contain one or more of them. The notion of a variant in this case follows the definition of the WebML pattern variants presented in Section 4.1. This way, we extract a first set of design constructs. Figure 1 depicts the retrieval of a design construct within a site view, while Figure 2 represents the identification of such a design construct retrieved from two site views of the conceptual schema. The constructs identified are stored in a *repository (of candidate patterns)*, along with their frequency and a list of other parameters described in Section 4.5.

Steps 4–6: Extending the sets of design solutions and their variants

- Following the procedure described in step 2 and taking into account the design constructs and variants’ definitions stored in the repository, we compute a new XML definition of each site view, by substituting – where possible – the variants with the default construct variant, aiming to create a more canonical schema. We then repeat

step 3 to mine a larger set of hypertext configurations that have not already retrieved in the previous step.

- Based on the technique introduced in Section 4.4, we try to capture larger – possibly combinations of – design constructs to enrich the repository with the maximum number of design solutions.
- We examine every pattern within the repository and in case it contains a predefined WebML content management pattern, but one or more of the remaining patterns is missing, we add the respective missing ones. For instance, if we locate a design construct containing a create pattern, we complement it by adding the modify and/or delete pattern(s).

Step 7: Design solutions evaluation and ranking

- It is obvious that the number of design solutions obtained by the above methodology can be very large if applied to a complex web application or even worse to a number of applications. Thus, a first evaluation of the candidate patterns has to be performed to decrease their number, and provide a first level of ranking. An analytical method accomplishing that is presented in Section 4.5.

Upon the completion of the above steps, the hypertext architect has access to a repository that contains a set of design solutions along with their variants. This library is composed of basic design configurations and combinations of larger ones that can be extended (in terms of usage) by defining new variants.

Figure 1 Retrieval of a predefined WebML pattern and a design construct within a site view

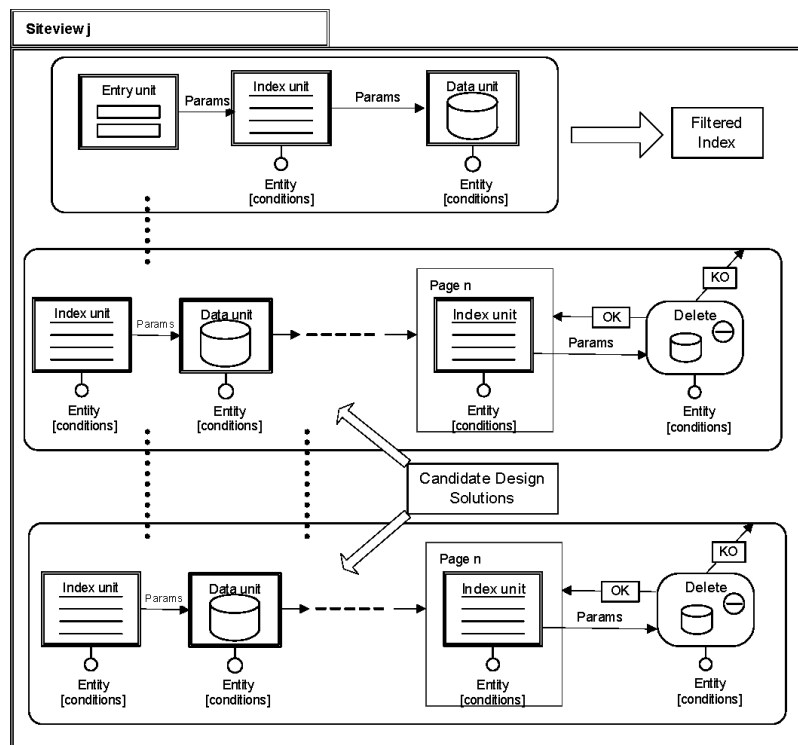
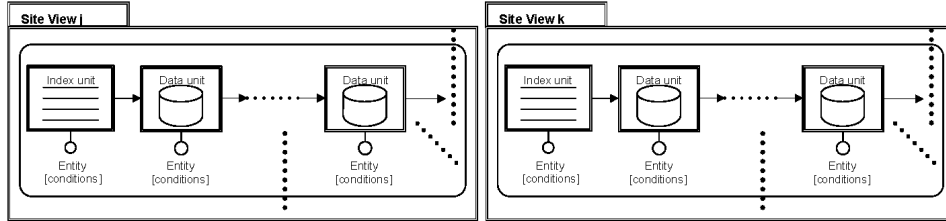


Figure 2 Retrieval of a design construct from different site views

As mentioned before, the application of the methodology to a variety of web applications can lead to the identification of templates for web application frameworks and patterns leading to effective, consistent and quality application development. More precisely, the designer based on the various design solutions and their variants retrieved by the methodology, if applied to a large number of applications in a specific domain (i.e., e-government), can form well-defined templates supporting the applications' commonalities and accommodating validations. Moreover, utilising languages such as *transformers-by example* (Lechner and Schrefl, 2004) (a language that allows defining transformers for WebML schemes by example¹) the designer can perform effective and consistent modelling activities in an automatic manner and construct wizards for future web application development.

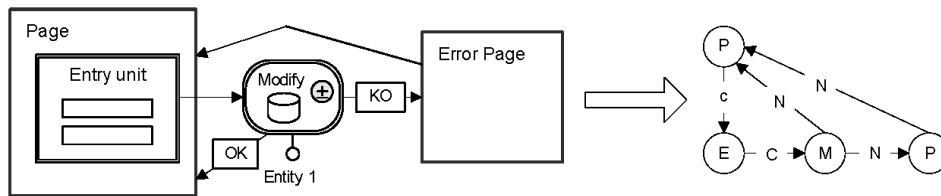
4.3 Automated extraction of design solutions

In this section, we describe a methodology for construct (compositions of hypertext elements) mining in the site views of a WebML fabricated web application. This approach is heavily based on graph mining algorithms. Intuitively, after modelling the site views as directed graphs, the task is to detect frequently occurring induced subgraphs. The problem in its general form boils down to finding whether the isomorphic image of a subgraph in a larger graph exists. The latter problem is proved to be NP-complete (Garey and Johnson, 1979). However, graph mining appears in many contexts including bioinformatics and chemistry, and therefore, quite a few heuristics have been proposed to face this problem. The most prominent approaches include *gSpan* (Yan and Han, 2002), *CloseGraph* (Yan and Han, 2003) and *ADI* (Wang et al., 2004). In the following, we provide some notation prior to reducing the problem to graph mining.

We define a site view as a directed graph, $G(V, E, f_V, f_E)$, comprising of a set of nodes V , a set of edges E , a node-labelling function $f_V: V \rightarrow \Sigma_V$, and an edge-labelling function $f_E: E \rightarrow \Sigma_E$. f_V assigns letters drawn from an alphabet Σ_V to the site view nodes, whereas f_E has the same role for links and the edge alphabet Σ_E . Σ_V has a different letter for each different WebML element, where 'element' includes content units, operations, pages, areas, etc. Correspondingly, Σ_E comprises of all the different kinds of links. We demand that units in WebML do not exactly correspond to nodes in V and the same is true for links between units and edges in E .

This choice was dictated by the rather complicated WebML conceptual model. We have to model the fact that a hyperlink can e.g. point to a data unit as well as to a hypertext containing several data units. Furthermore, links can also be classified into contextual and non-contextual, not to mention that a design construct can generally span different hypertexts. Therefore, before applying any graph mining technique, we preprocess the site view into its graph representation as follows: We process the WebML application definition and assign each unit and operation a letter according to its type. We install edges between units and label each edge with a 'C' (*contextual*), or with an 'N' (*non-contextual*). As a second step, we map each page, area, etc. into a separate node. An edge is introduced between e.g. a page-node and the nodes corresponding to elements it contains. This *containment edge* is labelled with a special letter 'c', to denote containment. Note that arbitrary containment sequences can exist. A transformation example is depicted in Figure 3.

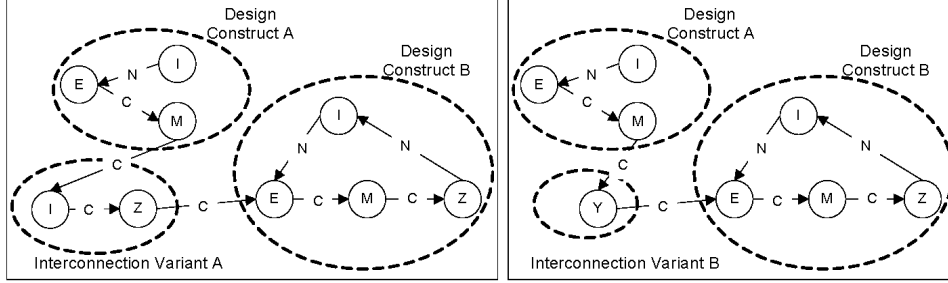
Figure 3 Transformation of a WebML pattern to its graph equivalent



Once having transformed all site views with the same methodology, design construct identification is reduced to mining frequent subgraphs of the site view database. The latter can be accomplished with any of the methodologies in Yan and Han (2002, 2003), and Wang et al. (2004) provided the desired *support* (the minimum percentage of occurrences in the entire database) is given.

4.4 A mechanism to acquire larger design solutions

A more complex case is the detection of implicitly interconnected design solutions. It is about the cases specified in step 5 of the proposed methodology. The main concept is that new broader design solutions appear when taking into account the intermediate constructs as parts of the already detected variants. A minimal approach would be to look for couples of constructs that are interconnected through a single link or unit passing context between the two constructs. However, we broaden the discovery procedure to include any number of constructs within a site view interconnected through intermediate structures. We think of larger design solutions to have identified constructs as a dominating part of them. As a result, we query for *interconnection variants* that are smaller than any other configuration involved in the larger design solution (Figure 4). Therefore, we utilise these constructs that include, at most, the minimum number of units among the configurations involved. In this way, we avoid exhaustive iterations or racing conditions. In the worst case, a whole site view would be evaluated as candidate design solution, but it will be rejected following the metrics given in the next section.

Figure 4 Acquiring larger design solutions

4.5 Design solutions evaluation metrics

To provide metrics in the design solutions' identification process, several evaluation factors are taken into consideration such as:

- *appearance* f denotes design solutions' frequency, which is comprised of:
 - overall number of appearing instances, f_o
 - appearance in f_α areas
 - appearance in f_σ site views
 - appearance in f_π pages
- *population* p denotes number of WebML elements involved defined as:
 - number of content units, p_δ
 - number of links, p_l
 - number of operation units (including generic operations defined by the designer), p_{op}
- *fragmentation* φ represents the number of pages hosting the design solution
- *entities involvement* e represents the number of data model entities participating in the design solution considered per conceptual factor as:
 - entities belonging to the core sub-schema, e_c
 - entities belonging to the personalisation sub-schema, e_{per}
 - entities belonging to the access sub-schema, e_{acc} .

We do not include the interconnection sub-schema because it refers to entities relations.

We define the notions of *importance*, *complexity* and *semantic value* as follows:

The importance V_j of the design solution j is:

$$V_j = f_j \times p_j \quad (1)$$

where

$$f_j = \frac{f_{o,j}}{f_{o,\max}} + \frac{f_{a,j}}{N_\alpha} + \frac{f_{\sigma,j}}{N_\sigma} + \frac{f_{\pi,j}}{N_\pi}, \quad f_j \in [0, 4]. \quad (2)$$

The notation f_{\max} is the maximum value of the corresponding metric and N is the corresponding sum of areas, site views and pages overall in a specific WebML definition.

$$p_j = \frac{P_{\delta,j}}{P_{\delta,\max}} + \frac{P_{l,j}}{P_{l,\max}} + \frac{P_{op,j}}{P_{op,\max}}, \quad p_j \in [0, 3]. \quad (3)$$

The complexity d_j of the design solution j is:

$$d_j = \frac{1}{\phi_j}. \quad (4)$$

The semantic value e_j of the design solution j is:

$$e_j = \frac{e_{c,j}}{e_{c,\max}} + \frac{e_{per,j}}{e_{per,\max}} + \frac{e_{acc,j}}{e_{acc,\max}}, \quad e_j \in [0, 3]. \quad (5)$$

Overall, the *impact* of a design solution j in a WebML conceptual schema is given by:

$$I_j = \frac{e_j}{\phi_j} V_j. \quad (6)$$

After computing the factor I_j for every design solution j in a specific conceptual schema, results are presented in a descending order to depict the most important and effective design solutions on the top. The same procedure can be applied repeatedly to site views from different applications to detect similar '*pattern*' implementation behaviours and group the results.

4.6 Evaluation of metric involvement value

Given the above metrics, we have used a single formula to combine them and calculate a single score for each pattern. However, in the case that one would like to utilise it in different governmental domains, then training data can be utilised to learn a regression model.

Regression is a classic statistical problem, which tries to determine the relationship between two random variables $x = (x_1, x_2, \dots, x_m)$ and y . In this case, the independent variable x can be just the vector of the four metrics proposed and the dependent y can be any real-valued score. We use y to sort candidate patterns in a descending order, thus the most important ones are shown on the top. Several regression models could be used, such as linear regression. We summarise linear regression below.

Linear regression attempts to explain the relationship of x and y with a straight line fit to the data. The linear regression model postulates that:

$$y = f(x) + \varepsilon = f(x_1, \dots, x_m) + \varepsilon \quad \text{or equivalently} \quad y = b_0 + \sum_{i=1}^m b_i x_i + \varepsilon \quad (7)$$

where the random error variable ε are iid normal with mean zero and variance σ^2 . The coefficients b_i ($0 \leq i \leq m$) are determined by the condition that the sum of the square residuals is as small as possible. Therefore, the linear combination with b_i should be better than those with any other coefficients. The variables X_i can derive directly from inputs, or some transformations, such as logarithm or a polynomial function, of inputs. When applying the proposed metric in different domains of e-government services, we can see that each property does not work very well independently and that different importance constants are computed in equation (7).

5 Exemplifying the methodology

Owing to space limitations, we will exemplify a fragment of the methodology, referring to an instance of a multinational governmental portal, in which we identify a candidate design solution. The data model of the application is rather simple and is omitted. We exemplify the third and sixth step of the methodology.

In the example depicted in Figures 5 and 6, we capture a candidate design solution, by traversing two distinct site views of the application’s conceptual schema. The first is a fragment of the public’s site view (Figure 5), depicting a *News* area (supporting categorisation of news) for the citizens and the second is a fragment of the organisation’s personnel site view including a *Forums* area (Figure 6), where the personnel can exchange messages in different categories of discussions.

Figure 5 A fragment of the public’s site view depicting the News section (*Get Country* is a *get unit* enabling the retrieval of a global parameter handling the applications multinational support)

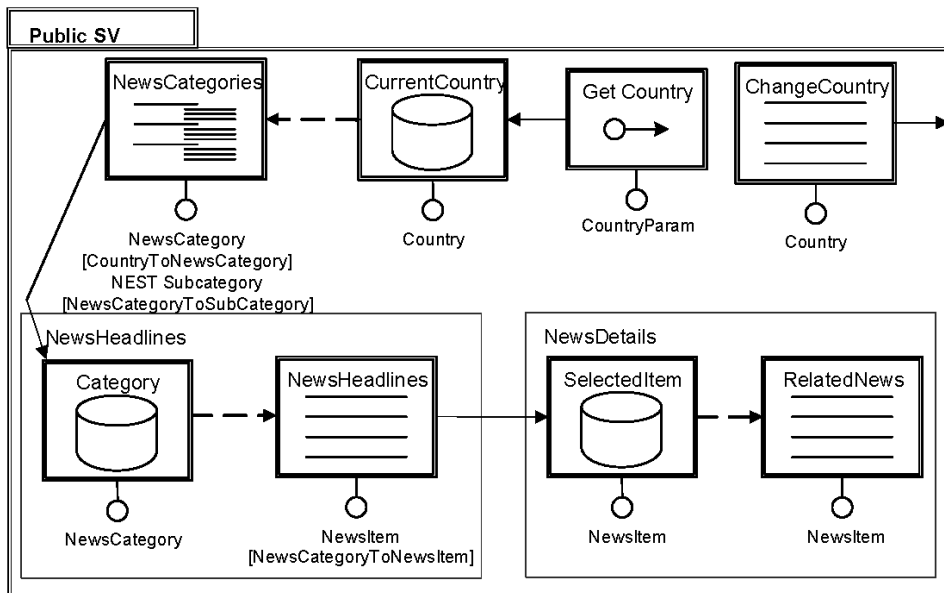
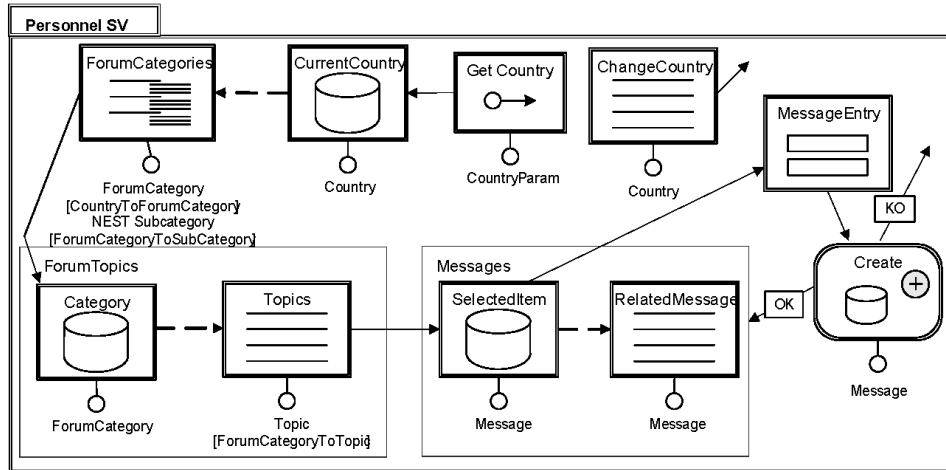


Figure 6 A fragment of the personnel's site view depicting a forum section

Comparing the two site views, we can easily identify the existence of a candidate design solution. It consists of the entire composition of WebML elements contained in the public site view. When applying the methodology, this design construct is stored in the repository. Moreover, in the case of the personnel's site view, one variant of the previously identified design solution can be extracted. This variant extends the design construct with an object creation pattern (as shown in Figure 6, the 'SelectedItem' data unit is linked with an entry unit used to supply values to a create unit).

Once variants containing content management patterns have been retrieved, we should extend the repository with variants derived by the missing content management patterns. For instance, a variant containing the object deletion pattern in place of the object creation pattern should be stored in the repository, as well as all the possible combinations computed using all the content management patterns.

Thus, although we have retrieved only one common navigation chain, we have constructed and stored in the repository more variants. Moreover, we have possibly identified a candidate design solution for the public information exchange within and outside the organisation.

6 Conclusions and future work

E-government enables both the augmentation in the quality of public services and an increase in cost-effectiveness in the service provision. The turnover expected when implementing government services in a web-based manner abound. Quicker access to the services can be achieved and input and other errors can be minimised during the procedures. E-commerce and e-learning paradigms successfully demonstrate the promising benefits. However, publishing simply information on websites in an ad hoc manner would not be translated into meaningful and successful e-government services. The challenge is made greater given the societal changes, which the EU will face over the coming years that make personalisation features unavoidable in all aspects of e-government services – both navigation, presentation and functional. Forecasts indicate

that in some Member States close to 40% of the population will be older than 65 years in 2020. As a consequence, without methodical design, careful evaluation of web-based e-government services starting at the early steps of development, a positive turnover cannot be ensured.

Aiming to increase the design and in second step the implementation quality of e-government services, this paper illustrated a methodology and provided metrics for discovering recurrent design solutions within a web applications' conceptual schema modelled using WebML. This methodology when applied to a large number of WebML application schemas can form the basis for the identification of templates in specific domain web application frameworks and become a valuable tool for hypertext architects for the identification of web design patterns.

The design solutions extracted can complement WebML predefined patterns in the process of modelling or redesigning an application providing effectiveness, reusability and consistency. Moreover, our methodology extends the quality evaluation framework presented in Fraternali et al. (2002, 2004) by means of providing a mechanism for capturing project data about the recurrent use of some design solutions, through the automatic analysis of XML application schemas.

In the future, we plan to extend the methodology by providing more precise metrics for the design solutions extraction taking into account a larger number of parameters quantifying the semantic context impact on the design configurations identified. Moreover, we plan to apply the methodology on a large number of web applications, to refine the methodology and fine-tune the design solutions evaluation metrics.

References

- Accenture Consulting (2001) e-Government Leadership Rhetoric vs. Reality – Closing the Gap.
- Alexander, C., Ishikawa, S., Silverstein, M., Jacobson, M., Fiksdahl-King, I. and Angel, S. (1997) *A Pattern Language*, Oxford University Press, New York.
- America's General Services Administration (2000) *Government and the Internet*, Survey.
- Bernstein, M. (1998) 'Patterns of hypertext', *Proceedings of HyperText '98*, Pittsburgh, pp.21–29.
- Booch, G., Jacobson, I. and Rumbaugh, J. (1998) *The Unified Modeling Language User Guide*, The Addison-Wesley Object Technology Series.
- Brambilla, M., Ceri, S., Comai, S., Fraternali, P. and Manolescu, I. (2002) 'Model-driven specification of web services composition and integration with data-intensive web applications', *IEEE Data Engineering Bulletin*, Vol. 25, No. 4, pp.53–59.
- Ceri, S., Fraternali, P. and Bongio, A. (2000) 'Web Modeling Language (WebML): a modeling language for designing web sites', *Proceedings of WWW9 Conference*, Amsterdam.
- Ceri, S., Fraternali, P. and Maristella, M. (2001) 'WebML application frameworks: a conceptual tool for enhancing design reuse', *Proceedings of WWW10 Conference Workshop on Web Engineering*, Hong Kong.
- Ceri, S., Fraternali, P. and Matera, M. (2004) 'Conceptual modeling of data-intensive web applications', *IEEE Internet Computing*, Vol. 6, No. 4, pp.20–30.
- Ceri, S., Fraternali, P., Bongio, A., Brambilla, M., Comai, S. and Matera, M. (2002) *Designing Data-intensive Web Applications*, Morgan Kaufmann, USA.
- Conallen, J. (1999a) 'Modeling web application architectures with UML', *Communications of the Association for Computing Machinery*, Vol. 42, No. 10, pp.63–70.
- Conallen, J. (1999b) *Building Web Applications with UML*, Addison-Wesley, Reading, MA.

- Demchak, C.C., Friis, C. and La Porte, T.M. (2000) 'Webbing governance: national differences in constructing the public face', in Garson, D. (Ed.): *Handbook of Public Information Systems*. Marcel Dekker Publishers, New York.
- eEurope 2005 (2002) *An Information Society For All*, Commission of the European Communities, Brussels, Available: http://europa.eu.int/information_society/eeurope/2002/news_library/documents/eeurope2005/eeurope2005_en.pdf
- eEurope 2010 challenges (2005) *European Information Society strategy up to 2010*, Available: http://europa.eu.int/information_society/eeurope/205/all_about/2010_challenges/index_en.htm
- eGov (2005) *The Official Web Site of the President's e-Government Initiatives*, Available: <http://www.whitehouse.gov/omb/egov/index2.html>
- Eschenfelder, K.R., Beachboard, J.C., McClure, C.R. and Wyman, S.K (1997) 'Assessing U.S. federal government websites', *Government Information Quarterly*, Vol. 14, No. 2, pp.173–189.
- Fernandez, M.F., Florescu, D., Kang, J., Levy, A.Y. and Suciu, D. (1998) 'Catching the boat with strudel: experiences with a web-site management system', *Proceedings of ACM-SIGMOD Conference*, pp.414–425.
- Fraternali, P. and Paolini, P. (1998) 'A conceptual model and a tool environment for developing more scalable, dynamic, and customizable web applications', *Proceedings of EDBT 1998*, pp.421–435.
- Fraternali, P., Matera, M. and Maurino, A. (2002) 'WQA: an XSL framework for analyzing the quality of web applications', *Proceedings of IWWOOST'02*, Malaga, Spain.
- Fraternali, P., Matera, M. and Maurino, A. (2004) 'Conceptual-level log analysis for the evaluation of web application quality', *Proceedings of IEEE LA-Web Conference*, Chile, pp.46–57.
- Gamma, E., Helm, R., Johnson, R. and Vlissedes, J. (1995) *Design Patterns – Elements of Reusable Object Oriented Software*, Addison Wesley, USA.
- Garey, M.R. and Johnson, D.S. (1979) *Computers and Intractability: A Guide to NP-Completeness*, Freeman, New York.
- Garzotto, F., Paolini, P. and Schwabe, D. (1993) 'HDM – a model-based approach to hypertext application design', *TOIS*, Vol. 11, No. 1, pp.1–26.
- Garzotto, F., Paolini, P., Bolchini, D. and Valenti, S. (1999) 'Modeling-by-patterns of web applications', *Proceeding of the ER'99 Workshop 'World Wide Web and Conceptual Modeling'*, Paris, France, pp.293–306.
- Golden, W., Hughes, M. and Scott, M. (2003) 'The role of process evolution in achieving citizen centered e-government', *Proceedings of the Ninth Americas Conference on Information Systems*, pp.801–810.
- Isakowitz, T., Stohr, E. and Balasubramanian, P. (1995) 'RMM: a methodology for structured hypermedia design', *Communications of the ACM*, Vol. 38, No. 8, pp.34–44.
- Lechner, S. and Schrefl, M. (2004) 'Transformers-by-example: pushing reuse in conceptual web application modelling', *Proceedings of 2004 ACM symposium on Applied computing*, Nicosia, Cyprus, pp.1654–1661.
- McClure, D.L. (2000) *Electronic Government: Federal Initiatives are Evolving Rapidly But They Face Significant Challenges*, Statement of David L. McClure, US General Accounting Office, before the Subcommittee on Government Management, Information and Technology, Committee on Government Reform, House of Representatives, Available: <http://www.gao.gov>
- Nanard, M., Nanard, J. and Kahn, P. (1998) 'Pushing reuse in hypermedia design: golden rules, design patterns and constructive templates', *Proceedings of ACM Hypertext'98*, Pittsburgh, PA, pp.11–20.
- Rainey, H.G. (1979) 'Perceptions of incentives in business and government: implications for civil service reform', *Public Administration Review*, Vol. 39, No. 5, pp.440–448.

- Rainey, H.G. and Bozeman, B. (2000) 'Comparing public and private organizations: empirical research and the power of the a priori', *Journal of Public Administration Research and Theory*, Vol. 10, No. 2, pp.447–470.
- Rainey, H.G., Backoff, R.W. and Levine, C.H. (1976) 'Comparing public and private organizations', *Public Administration Review*, Vol. 36, No. 2, pp.233–244.
- Schwabe, D. and Rossi, G. (1998) 'An object-oriented approach to web-based application design', *Theory and Practice of Object Systems (TAPOS)*, Vol. 4, No. 4, pp.207–225.
- Schwabe, D., Esmeraldo, L., Rossi, G. and Lyardet, F. (2001) 'Engineering web applications for reuse', *IEEE Multimedia*, Vol. 8, No. 1, pp.20–31.
- Schwabe, D., Garrido, A. and Rossi, G. (1997) 'Design reuse in hypermedia applications development', *Proceedings of ACM Hypertext '97*, Southampton, UK, pp.57–66.
- von Dran, G.M., Zang, P. and Small, R. (1999) 'Quality websites: an application of the Kano model to website design', *Proceedings of the 5th Americas Conference in Information Systems-AMCIS'99*, pp.898–900
- Wang, C., Wang, W., Pei, J., Zhu, Y. and Shi, B. (2004) 'Scalable mining of large disk-based graph databases', *Proceedings of ACM KDD04*, pp.316–325.
- Wang, L., Bretschneider, S. and Gant, J. (2005) 'Evaluating web-based e-government services with a citizen-centric approach', *Hawaii International Conference on System Sciences-38*, Kona, Hawaii.
- WebRatio (2005) Available: <http://www.webratio.com>
- West, D. (2000) *Assessing E-Government: The Internet, Democracy, and Service Delivery by State and Federal Governments*, Brown University, Available: http://www.brown.edu/Departments/Taubman_Center/polreports/egovtreport00.html
- World Markets Research Centre (2001) *Global E-Government Survey*.
- XSL (2001) *Extensible Style Sheet Language*, W3C Recommendation, <http://w3.org/TR/XSL/>
- Yan, X. and Han, J. (2002) 'gSpan: Graph-based substructure pattern mining', *Proceedings of Int. Conf. on Data Mining (ICDM'02)*, Maebashi, pp.721–724.
- Yan, X. and Han, J. (2003) 'CloseGraph: mining closed frequent graph patterns', *Proceedings of ACM-KDD03*, pp.286–295.

Note

¹I.e., by giving an example of what is desired instead of specifying operations for achieving the result.