Designing Websites with EXtensible Web (*x*Web) Methodology

RAJUGAN, R., WILLIAM GARDNER, THARAM S. DILLON
eXel Lab, Faculty of Information Technology, University of Technology, Sydney, Australia
Email: add

ELIZABETH CHANG
School of Information Systems, Curtin University of Technology, Perth, Australia

Received: July 6 2005; revised: October 8 2005

Abstract— **Today, eXtensible Markup Language (XML) is fast emerging as the dominant standard for storing, describing, representing and interchanging data among various enterprises systems and databases in the context of complex web enterprises information systems (EIS). Conversely, for web EIS (such as e-commerce and portals) to be successful, it is important to apply a high level, model driven solutions and meta-data vocabularies to design and implementation techniques that are capable of handling heterogonous schemas and documents. For this, we need a methodology that provides a higher level of abstraction of the domain in question with rigorously defined standards that are to be more widely understood by all stakeholders of the system. To-date, UML has proven itself as the language of choice for modeling EIS using OO techniques. With the introduction of XML Schema, which provides rich facilities for constraining and defining enterprise XML content, the combination of UML and XML technologies provide a good platform (and the flexibility) for modeling, designing and representing complex enterprise contents for building successful EIS. In this paper, we show how a layered view model coupled with a proven user interface analysis framework (WUiAM) is utilized in providing architectural construct and abstract website model (called eXtensible Web, *x*Web), to model, design and implement simple, user-centred, collaborative websites at varying levels of abstraction. The uniqueness *x*Web is that the model data (web user interface definitions, website data descriptions and constraints) and the web content are captured and represented at the conceptual level using views (one model) and can be deployed (multiple platform specific models) using one or more implementation models.**

Index Terms— **OO conceptual models, web engineering, web user interface design, XML-views, WUiAM, XML**

I. Introduction

In software engineering, many methodologies have been proposed to capture real-world problems into manageable segments, which can be communicated, modelled and developed into error-free maintainable software models and modules [1][5]. Similarly, in the case of Web Engineering, both web content and specifications should represent meaningful units of information with respect to the semantics of the domain in question [6][7].

Web engineering and website development have evolved from coding simple HTML based static pages into complex a software engineering discipline. The traditional web engineering techniques, which were based around textual file based structures, provided only limited or no facilities for modeling higher-level design concepts that go beyond the granularity of file-based textual information [8][9][2][4]. But, today's websites not just deliver static contents, but also support (web) application driven data transactions to complex multimedia web contents [3][10][11]. Thus with complex web contents such as interactive and hypermedia based web sites, designers went beyond traditional HTML files and turned towards middleware and scripting technologies, from Common Gateway Interface (CGI) scripts to advanced SOAP messages. While there are speedy advancement in the implementation and deployment level technologies, there is a lack of sufficient techniques with modeling and design capabilities that are available to classical (non-web based) software solutions. One of the most common architecture for the deployment of such web system is the 3-tier architecture, which consist of the presentation layer, application layer, and data layer.

In designing today's websites, there are many challenges faced by web engineers. These challenges may be grouped into; (i) nature of web technologies (in comparison to standalone software solutions), (ii) user interface design constraints and (iii) web content (traditional structured data vs. unstructured data). Due to some of the unique characteristics of the web such as; (a) unknown end-user base (for e.g. number of users, skill-level, user-training, etc), (b) untested or targeted end-user skills, (c) unstructured and heterogeneous content formats that are mainly presentation oriented, (d) stateless processes and (e) unknown end-user/client platform and configuration, made it almost impossible to develop a uniformed single web engineering technique that is acceptable all. Therefore, since the beginning of the dot.com boom, researchers have proposed various web engineering techniques and methods to make web engineering an achievable dream.

In the case of user-interface engineering, due to the unique characteristics of web architecture, there is a great degree of flexibility in the design of the Web User Interface (WUI) [12][13], compared to the user interface design of traditional software applications. This in turn has had some undesirable consequences. Like the development of any software application, a systematic approach is required to support the design process of WUI; hence the application can support the users to complete their tasks efficiently and effectively. The process of WUI design involves both logical and perceptual aspects. Without careful analysis of the UI requirements, usability problems can result in the implemented application. The process of WUI design involves both logical and perceptual aspects [14].

Web standards are one of the issues that define how web technologies can progress to new levels. Similar to the importance of web engineering methodologies, it is also important to have uniform web standards (such as protocols, languages such HTML, XML etc) that are capable of handling heterogeneous web content, but also support the emerging new engineering processes. To address such issues, in late 90's, many web engineering approaches namely; (a) Objected-Oriented (OO) for web content engineering using portable web languages and databases, (b) relational DBMS/middleware powered web engineering (e.g. later Oracle Portal) and (3) combination of both (a) and (b). But, to best of our knowledge, none of the approaches provide a comprehensive, yet generic solution that helped web engineering from conceptual modeling to implementation.

Conversely, as industrial production techniques move towards a distributed model, the need to exchange data via web, between heterogeneous data sources [15][16] is constantly increasing. These heterogeneous data sources could arise from server groups from different manufacturers or databases at different sites with their own schemas. However, since the introduction of eXtensible Markup Language (XML) [17], it is fast emerging as the dominant standard for storing, describing and interchanging data among various Enterprises Information Systems (EIS) and heterogeneous databases. In combination with XML Schema [18], which provides rich facilities for constraining and defining XML content, XML provides the ideal platform and the flexibility for capturing and representing complex data formats structures, including meta-data for web contents.

In a typical enterprise website, web contents may range from static un/semi-structured textual data to binary multimedia streams and dynamic on-the-fly hypermedia contents. Another dimension to web content is that, they are distributed and hosted by multiple geographically distributed servers and databases. In direct contrast to DBMS managed structured data (relational or Object-Relational or OO), the web content do not conformed to traditional data models which assume a fixed data model/schema for a given set of data domain. Therefore since the introduction of Internet and the World-Wide-Web, researchers, standard organizations and the Industries rallied around to adopt a web data language that is semantically rich and descriptive yet conforms to open standard schema, which can support and describe all types of data and content on the web, including the traditional structured data.

XML documents which are tag-based and self-describing data documents that represent a hierarchical tree structure. At the conceptual level, they can be visualized as hierarchical trees or graphs. An XML document is usually associated with a Document Type Definition (DTD) or XML Schema which is used to define and constrain the syntax and structure of a document. XML schema provides more rich facilities for descriptive user-defined elements and attributes specification, with the flexibility of re-use and flexible constraint definitions. XML schema, which itself is a XML document can be also represented as a hierarchical tree/graph structure. Though semantically descriptive, XML document and its associated schema provides less human comprehensible model in comparison to diagram based models such as UML. Most of these challenges are addressed in proposing the *x*Web, described in the following sections.

One other interesting development is the introduction of OMG's Model Driven Architecture (or MDA) [19], which presents an interesting OO based paradigm for Industrial Informatics and system modelling; that is, in MDA, specifications of the system operations are separated from the details of the platform/implementation specific syntax or specifications. For such approach to be successful, under MDA, all models have to be specified in an orderly manner, at a higher-level of abstraction, which in turn should be easily mapped to platform specific specifications. Therefore it is interesting to look at web solutions under the MDA initiatives, with utilization of XML technologies for web content management.

In MDA, platform independent models play a vital role in system development and data engineering. Under the MDA initiative, first the model of a system is specified via an abstraction notation independent of the technical or deployment specifications (i.e. Platform Independent Model or PIM) and then the PIM is mapped or transformed into a deployment model (i.e. Platform Specific Model or PSM) by adding platform or deployment specific information into the PIM. To support MDA initiatives in data engineering, data semantics, constraints and model requirements has to be specified precisely at a higher level of abstraction. This presents an opportunity to investigate data views as a means of providing data abstraction and semantics in PIMs for data intensive MDA solutions.

In the context of web engineering solutions under MDA, it is still a challenging task to produce PIMs. This is mainly due to OO modeling languages such as OMG UML [20], Extended-ER [21] etc. provide insufficient modeling constructs for utilizing heterogenous schemas (e.g. XML, RDF-S) and web content descriptions and constraints, while many web language schema (CSS, HTML, XML, RDF etc.) lacks the ability to provide higher levels of abstraction (such as conceptual models, visual constraints) that are easily understood by humans.

In this context, to this date, many web engineering solutions (for simple web site design to e-Commerce engineering) are focused mainly on building, designing and maintaining web pages (using various styles and techniques) to support user-requests (from displaying simple text information to B2B and B2C transactions) without consideration for all four combined aspects of web engineering namely;

- a well defined design methodology (such as OO) using standard modeling language (such as OMG's UMLó [20]) to capture and model (a) domain requirements,

(b) user requirements, (c) user-interface requirements (user-interface engineering), (d) data requirements (data engineering) and (e) platform/architecture requirements,

- a generic architecture adoptable to various implementation techniques,
- well defined generic data standard (such as XML) to describe and represent web data and
- a process to address post-development maintenance and expansion.

Though there exists tools that can perform one or two of the above stated aspects in regard to web engineering, to our knowledge no tool or techniques provide all four aspects. In this paper we propose web engineering methodology (called eXtensible Web or *x*Web) [22] that we argue will provide all four aspects of the software engineering process for a comprehensive web engineering methodology. The *x*Web is an architectural construct and a design methodology based on; (i) Object-Oriented conceptual modeling (OOCM) principles, (ii) Web User Interface Engineering using the Web User Interaction Analysis Model (WUiAM) [12][23] and (iii) web content abstraction using a Layered View Model for XML [24].

The rest of this paper is organized as follows. In section 2, we briefly look at some early work done in relation to website engineering, followed by section 3 which describes and defines some of the core technologies used in *x*Web. Section 4 outlines *x*Web in detail, including its components and the *x*Web architecture. Section 5 describes a detailed walkthrough of the *x*Web design steps, including domain model, use-case analysis and the WUiAM model, using a case-study used to illustrate the *x*Web concepts. Section 6 concludes the paper with some discussion on our future research directions.

II. Related Work

Here we look at some work done in web design approaches and in particular, website and web portal design, as our work incorporates both. There exists many work and tools in dealing with one or more aspects of general web engineering principles, works such as [6][8][14][25][26], but none of these address the whole spectrum of web engineering issues. Also, there are many existing works that deal with the possibility of application of portal in different areas of use [27], and the classification and discussion of different type of portal [13][28]. Only few have look into the issue of the actual design and development of a portals. One of the most interesting work includes [29], where the authors looks at the development of portal from a software engineering perspective. In [30], usability issues are taken into account and the importance of evaluating these on customizable portals is also discussed. But most of these works do not provide a comprehensive design and technological solution for addressing both web data and web user interface design issues under one design methodology. We argue that, such a combined design methodology is a must for any web system development such as re-usable websites and portals.

In [31], the authors have argued that there are two aspects of technical architecture that a web modeling language must possess for it to be used effectively on the development of web systems, namely information architecture and functional architecture. In the related literature, there is a lack of consideration to the idea that the implementation of a web user interface (WUI) is quite difference to that of a traditional software system, as traditional software GUI is mainly constructed through the use of GUI widgets. Also, the kinds of device that are used for the display of WUI are much diverse, such as PDA's, mobile phone, etc. Over the years, several techniques have been introduced in the literature for the modeling and design of web-based systems. There is a heavy concentration in the earlier methods to be; (1) hypertext oriented [32][33], or (2) data centric or data driven [26]. While some of the more recent methods have its base on (3) object oriented paradigm [6]. These models were found to not pay sufficient attention to users, who are central in web systems. These systems, hypertext, or data centred approaches need to be contrasted with the (4) User-Centred approach [23][25][34].

An interesting new development in web engineering is the utilization of ontologies. Works such as [9][10][16][35][36][37] looked at the problem of web engineering (and web portal) from an ontology engineering point of view and some works have successful implementations. Though this is a good problem to solve from a domain-specific website engineering (such as the GONG project, http://gong.man.ac.uk/), we argue that, using ontologies to model and design generic and/or domain independent websites are time-consuming and complex. Designing an ontology base for modelling user interaction/profile is itself a complex and ongoing research problem [9][23][27] and combing generic website semantics into such an ontology base will increase size and complicity of the ontology base beyond practical use, as complex query processing and sub-ontology extraction algorithms are required to construct the website pages.

In general, many of more recent web design methods attempt to address navigational design partly in certain way within the proposed process. However, the navigational model is often a by-product of the underlining domain model, which does not always provide the user view required as the user would like to perceive the information. Rather, it had only map this data model that are a suitable representation of the data for storage and efficient for system manipulation directly onto the presentation layer. It can be observed there is the assumption that all data source come form the internal system. However, with the swift advent of technologies such as web services, agent-base system, the final contexts that are presented to user on client device may include content from a number of different data sources. This will certainly have fundamental effect on the way how the whole system is to be built.

III. Preliminaries

Our approach to web engineering is a Conceptual View driven User Centric architectural construct and a design methodology, with extensive support for; (i) web content abstraction and meta-data using the layered view model and (ii) web user interaction analysis and design using WUiAM, for modeling and building user-centric, content-based websites

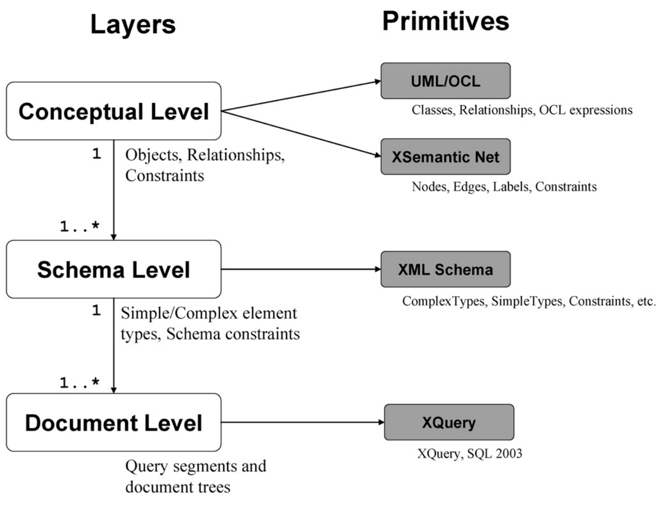

Layers
Primitives
Conceptual Level
UML/OCL
Classes, Relationships, OCL expressions
XSemantic Net
Nodes, Edges, Labels, Constraints
1
Objects, Relationships, Constraints
1..*
Schema Level
XML Schema
ComplexTypes, SimpleTypes, Constraints, etc.
1
Simple/Complex element types, Schema constraints
1..*
Document Level
XQuery
XQuery, SQL 2003
Query segments and document trees

Fig. 1. The Layered View Model for XML (context diagram).

(and web portals [38]). The uniqueness *x*Web is that the model data (web user interface definitions, website data descriptions and constraints) and the web content are captured and represented at the conceptual level using views (one model) and can be deployed (multiple platform specific models).

To understand *x*Web, in is imperative that, one should understand both the layered view model and the WUiAM in some detail. Here, in this section, we briefly describe these models in details.

A. The Layered View Model

The layered view model [24]used here has three-layers of abstraction (Fig. 1), namely; (1) conceptual, (2) logical or schematic and (3) document or instance level. The view model is based on the postulates 1 and 2, about the real world.

Postulate 1: The term context refers to the domain that interests an organization as a whole. It is more than a measure and implies a meaningful collection of objects, relationships among these objects, as well as some constraints associated with the objects and their relationships, which are relevant to its applications.

It should be noted that, the context here refers to static metadata and web content specific context and not to be confused to navigational context described in section III.B(3) below.

Postulate 2: The term view refers to a certain perspective of the context that makes sense to one or more stakeholders of the organization or an organization unit at a given point in time.

1) Conceptual Views: The conceptual layer (Fig. 1) describes the structure and semantics of XML views in a way which is more comprehensible to human users. It hides the details of view implementation and concentrates on describing objects, relationships among the objects, as well as the associated constraints upon the objects and relationships. Due to its abstract nature, conceptual views can be defined using any high level modeling languages such as Dillon and Tan notation [5], UML [39], XSemantic Nets [40] or Extended Entity-Relationship Model (E-ER) [21]. The conceptual views are constructed using a set of conceptual operators [41].

Definition 1: A conceptual view is the one which is defined at the conceptual level with higher level of abstraction.

One such abstract view formalism will;

i provide data abstraction to view data set similar to a class (in OO) does to real-world objects,

ii enable the software designers (not the programmers) to visualise, construct and validate constructed data sets (views) that are normally left to implementers,

iii utilise as a tool to communicate better with the domain users (DU) and to improve domain user feedbacks (as DU usually used to visualise data as a constructed data sets (views) than a stored/modelled data class),

iv be utilised in other areas, such as User Interface Engineering (UIE), where abstract constructs can be constructed at the conceptual level to capture Abstract User Interface (AUI) objects [42], where the user interface objects are identified based on what the user interface does and not how it is done and

v be utilised by system designers to add additional data semantics at a higher level of abstractions to data intensive domains (such as XML based domains), where the meaning of data is important than the data itself.

In related literature, the notion of conceptual views is non-existent. From relational to semi-structured and XML, the view concept begins at the data manipulation language level.We argue that, providing view formalism at the conceptual level (abstract views) will improve the resulting view implementation similar to that of a conceptual model what does to a software system.

First we briefly explain some of the terms used in the layered view model.

Conceptual Objects (CO): CO refers to model elements (objects, their properties, constraints and relationships) and their semantic inter-relationships (such as composition, ordering, association, sequence, all etc) captured at the conceptual level, using a well-defined modeling language such as UML, or XSemantic nets or E-ERD [21], etc. A CO can be either of type simple content ($s_{content}$) or complex content ($c_{content}$) depending on its internal structure [43] [44] [45]. For example, CO that uses primitive types (such as integer, character etc) as their internal structure corresponds to ($s_{content}$)and CO that uses composite objects represent their internal structure corresponds to ($c_{content}$).

Conceptual Schema (CS): We refer conceptual schema as the meta-model (or language) that allow us to define, model and constrain COs. For example, the conceptual schema for a valid UML model is the MOF (combined with its associated MOF meta-model elements such as stereotypes and data dictionaries). Also, the UML meta-model provides the namespace of such schemas.

Like XML Schema, where the instance will be an XML document, here, an instance of the conceptual schema will be a well-defined, valid conceptual model (in this case in UML) or other conceptual schemas (ie. meta-model such as MOF), which can be either visual (such as UML class diagrams) or textual (in the case of UML/XMI models).

Logical/Schema Objects (LO): When CO are transformed or mapped into the logical/schema level (such rules and mapping formalism described in works such as [43] [45] [46]), the resulting objects are called SO. These objects are represented in textual (such as a schema language) or other formal notations that support schema objects (such as graph). In our work, for logical/schema level we use XML Schema as the schema model language.

Thus, a *context* (ζ) may be stated as an item (or collection of items) or a concept that is of interest for the organization as a whole. It is more than a measure [47] [48] and is a meaningful collection of model elements (classes, attributes, constraints and relationships) at the conceptual level, which can satisfy one or more organizational perspective/(s) in a given domain. Also, a *perspective* (ϑ) may be stated as a viewpoint of an item (or a collection of items) that makes sense to one or more stakeholders of the organization or an organizational unit, at given point in time. That is, one viewpoint of a context at a given point in time.

Definition 2: A conceptual view (V_{co}) is a view, defined over a collection of valid model elements, at the conceptual level. That is, it is a perspective in a given context at a given point in time.

A valid conceptual view of the valid CO collection set χ is defined as the perspective (ϑ) constructed over a context (ζ) by the conceptual construct λ [41]. The resulting conceptual view belongs to the domain $dom(V_{co})$, (where $dom(V_{co})$ = $dom(\zeta)$) with schema $S_{co}(V_{co})$, (where $S_{co}(V_{co}) = S_{co}(\vartheta)$). The conceptual view is said to be valid if it is a valid instance of the view schema. Let V be a function of a view, therefore conceptual view V_{co};

$$V_{co} = V(\zeta,\ \vartheta,\lambda,\ \chi)$$

where, the *view name* of V_{co} is provided by the perspective ϑ; the namespace for V_{co} is provided by the *context* ζ in the valid CO collection set of χ; the view construction is provide by the conceptual construct λ (i.e. *conceptual operators* that construct the view over a given context); the valid collection set χ provides the data for the view instantiation; the *view schema* is provided by $S_{co}(V_{co})$ which constrains and validates the view instances of the view V_{co}, and the domain is provided by $dom(V_{co})$ for the view V_{co}.

2) Logical Views: The middle schema (or logical) layer (Fig. 1) describes the schema of XML views for the view implementation, using the XML Schema language. Views at the conceptual level are mapped into the views at the schema level via the transformation mechanism developed in work [43] [45] [46]. The output of this level will be in either textual (such as XML Schema language) or some visual notations that comply from the schema language (such as graph).

Definition 3: A *logical view* is an imaginary (XML) document schema which points to a collection of semantically related (XML) tag definitions from a domain and satisfies a conceptual view definition from the target conceptual domain.

To continue from our discussion from conceptual views, we have stated that, a *logical/schema object (LO/SO)* is such that; when a conceptual object (*CO*) is transformed or mapped into the logical/schema level (such rules and mapping formalism described in works such as [43] [45] [46] [49] [50]), the resulting objects are called *LO*. These objects are represented in textual (such as a schema language) or other formal notations that support schema objects (such as graph). In our work, for logical/schema level we use XML Schema as the schema language.

Let $\aleph_{lo}^{co}$ denotes a schemata transformation of valid CO (e.g. in UML) into LO (in XML Schema elements [45] [49] [50] with added extension for conceptual views [51]. Also, let $\aleph_{lo}^{co}$ (X) be a generic transformation rule set that transforms a valid conceptual collection into a valid logical collection, where "**co**" indicates the collection of CO and "**lo**" indicated the target LO/SOs (in our case, the XML Schema (xsd)). This can be shown as;

$$X^{\wedge} = \aleph_{XSD}^{UML}(X)$$

where $X^{\wedge}$ is the valid logical collection of the valid conceptual collection X.

Definition 4: A logical view (V_{lo}) is a view, defined at the logical/schema level, which satisfies a conceptual view, with a valid view definition, a valid view schema, and a query segment to construct the view from the stored collection of documents.

Therefore, we can show our logical view (V_{lo}) of a valid conceptual view represented in UML as;

$$V_{lo} = V_{co}^{\wedge} = \aleph_{XSD}^{UML}(V_{co})$$

The valid (XML) collection set is provided by the instances of X for the view V_{lo} instantiation. The data extraction is provided by the transformation;

$$\lambda_{query} = \aleph_{XSD}^{UML}(\lambda)$$

The view schema $S_{lo}(V_{lo})$ that constrains and validates the view instances of the view provided by the $S_{lo}(V_{lo})$ = $\aleph_{XSD}^{UML}(S_{co}(V_{co}))$. Here, in our work, the logical schema is XML Schema [18].

The domain provides the domain for the view resulting from the transformation.

It should be noted that, the relationship between V_{co} and V_{lo} is one-to-many [45] as one UML CO collection can be mapped to one or more valid XSD LO collections.

3) Document Views: The document (or instance) level (Fig. 13) implies a fragment of instantiated XML data, which conforms to the corresponding view schema defined at the upper level.

Definition 5: A document (or instance) view is an XML document that is instantiated or materialized (at the document level), where the instance is well-formed and valid document (here XML) that conforms to a corresponding logical view.

The instantiation is done by a document view query expression (here native XML query language).

Also, we can state this more formally, as;

Definition 6: An imaginary (XML) document (V_{doc}) is a logical view that is instantiated and/or materialized (at the document level), where the instance is well-formed and valid document (here XML) with view schema $S_{XSD}(V_{lo})$.

The V_{doc} is instantiated or constructed by a query segment (native XML query language syntax) or specific algorithms such as OEA [52]. This transformation is shown by $\lambda_{query} = \aleph^{UML}_{XSD}(\lambda)$. In order to keep the scope of this paper focused on conceptual and logical view extensions, we do not provide a detailed discussion on this transformation methodology.

In the case of *x*Web, the conceptual view model mentioned above is equivalent of the PIMs in the MDA framework. The logical views and the corresponding document views can be considered as the (static) PSM models of the corresponding conceptual views. A detailed discussion on this layered view model and formal properties can be found in [24] [41].

B. Web Interface Dynamic Analysis Modeling

The process of WUI design involves both logical and perceptual aspects. Without careful analysis of the user interface UI requirements, usability problems can result in the implemented application. For this task, *x*Web is support by the use of the Web User Interaction Analysis Model (WUiAM), which captures the user interaction for WUI at a higher level of abstraction. WUiAM is a systematic approach that allows the specification of an analysis model based on a task/activity oriented approach. It is a modelling method for representing the possible user-system interaction requirements. The information captures in WUiAM should be isolated from any specific visual or graphical design concerns; it gives a logical view of the Web User Interface (WUI) that is being considered. This helps to ensure the accuracy and completeness of a user view definition, and helps clarify the requirements for visual design of the actual WUI at a later stage of the development.

The complete WUiAM contains two layers (Fig. 2), namely Context Navigation Net (CNN) and In-context Flow of Interaction Net (iFIN). CNN, the purpose of this top layer of the model allow the overall depiction of user movement within the complete system domain to be capture as a whole be on user task analysis, this process will be further discussed in section 5.3 with the illustrated case study example. Essentially, two types of user interaction behaviors are captured by the navigational context on the CNN layer:

i User to System – A major task/activity that a user is required to complete in order to achieve a particular goal.

ii System to User – A response by the system to the user as a result of some system process that is triggered by user actions.

Transition between navigational contexts as defined in the CNN (The complete set of notation of CNN is shown in Fig. 3) model can be of two types:

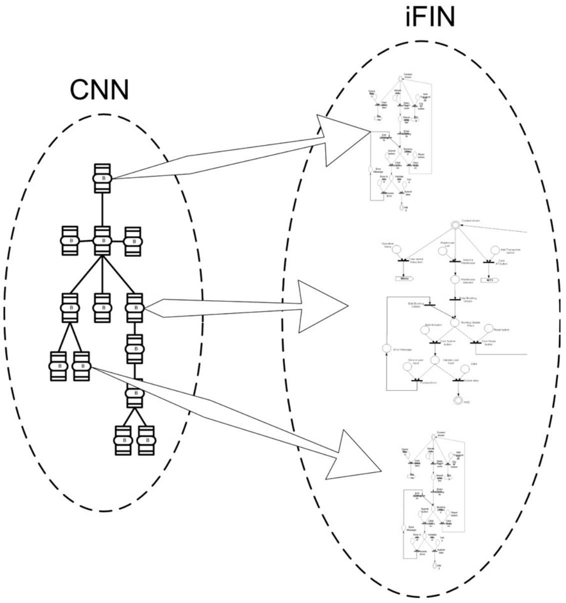


Fig. 2. WUiAM – illustration of the two-layer concept.

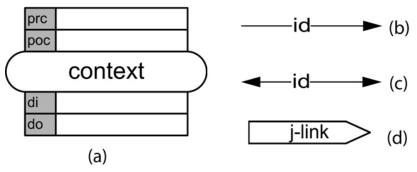


Fig. 3. Context Navigational Net notations.

i Unidirectional sequential transitions: A sequential flow of control focus from the current context to the next context is represented as a solid line with a feathered arrow pointing to the next context.

ii Bidirectional sequential transitions: A double ended arrow represents the possibility of bidirectional transition between contexts.

iii Non-sequential transition: Jump link from the current context. A Jump link is defined as a possible switch in context where the destination context are a group in a defined navigational structure, or a single destination that are consider of non-sequential nature based on the current path of the U_{Task} in progress.

The iFIN, on the other hand, provide a method for the magnification of user interaction within a particular context. iFIN is a form of Perti Net and it consists of a set of graphical notation (based on the Perti Net) proposed for the logical design and modelling of user interface. The main constructs of iFIN are shown in Fig. 4. These include a) state, b) action, c) transition, d) start/end state, e) jump-link, and f) token. In iFIN, a state is a representation of the state of an Abstract Web User Interface class UI^{W}_{abs}, a state before the user action de-notes the pre-condition, and the one after denotes the post-condition in the current user interaction progress. Note here this pre-condition and post-conditions are of the micro level condition internal to the current active navigational context, and therefore different to what is defined on the CNN level. A token appears in the state shown, indicate that the current

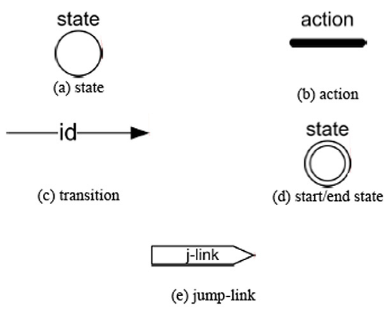


Fig. 4. In-context Flow of Interaction Net notations.

command object is activated, and when all places before a user action are activated, the user action can be fired, and hence cause the transition to occur. A powerful feature of iFIN is its ability to expend and shrink part of the model when required. The meaning of this is that the model can defined down to the very details of all possible UI constructs, or certain details can be collapse into smaller section, while the model still maintain its semantic. The complete set of notation of iFIN is shown in Fig. 4.

The websites in the *x*Web is designed based on User Tasks (U_{Task}). For example, a user responding to a conference call-for-paper, who visits the conference website may have following user tasks, namely (a) gather what is the aim of the conference, (b) check important dates and (c) hopefully submit a paper. An online shopper visiting Amazon website may have the following user tasks, (a) check new releases, (b) check discount offers, (c) place an order and/or (d) add items to wish list. The concept of User Task in *x*Web can be defined base on the notion used in the Web User interaction Analysis Model [12][23].

Definition 7: An User Tasks (U_{Task}) is a series of interactions performed by a web user in order to achieve his/her main task or goal.

A user task U_{Task} defines a possible set of context, but a single user task it may include the complete set of context that can be defined for a system. The complete set of user task can be described using open-ended net such as CNN. Contexts are connected by transitions.

$$U_{Task} \subseteq \text{CNN}$$

Definition 8: A **navigational context** C_{nav} is a perspective over a collection of logically related user interface object Obj_{UI} that are perspective composite model from the conceptual level, such that a context corresponds to an Abstract Web User Interface class [53] - UI^{W}_{abs}, user interaction that occurs within the navigational context and it will result in a change of business state in the system. Each navigational context is a representation of the user perspective U_{view} of a collection of logically related objects. This can be translated into the representation of an Abstract User Interface class UI_{abs} is a selection or a projection or other user-defined operation on one or more underlying domain classes and detail internal dynamic property of a navigational context is defined by its iFIN.

The high level dynamic of a navigational context is characterizes by four attributes:

- Pre-conditions are conditions that need to be met before a context can become activated, this helps the developer to see the kind of access control that will be required for each context.
- Post-conditions are the conditions that must be met before control will be passed onto the next context, these conditions will guard the flow of sequential transition. However, for non-sequential transition these conditions will not need to be met.
- Data Input is data that must be passed into the current context from the previous sequential context, if any value is present for this object, it means that a jump link entry to this context will also need to meet this requirement.
- Data Output is data that will traverse into the next sequential contexts, again for the situation of non-sequential transition, this condition will not need to be satisfied.

Definition 9: User action U_{action} would cause the transition that result in the activation and deactivation of the current navigational context. A context is a generalization of the notion of Activity in Activity Diagram. Features that characterize a context are pre-condition, post-condition, data input, and data output.

As Obj_{UI} can be classified into two type; namely Obj^{cmd}_{UI} and Obj^{inf}_{UI}. The following Postulates describe these in detail.

Postulate 3: Obj^{cmd}_{UI} provide the following facilities; (i) trigger navigation from one context to another context; (ii) trigger portion of the UI to change accord to user action; (iii) moving data (i.e. trigger the movement of data from the UI layer to the application program or database or vice versa, or in between UI objects); (iv) initiating an action within the system or stopping the action being carried out by the system.

Postulate 4: Obj^{inf}_{UI} provide the following facilities: Information Objects have only static property. Its sole purpose is a container for representation of a collection of meaningful data and for the display this data. They are essentially a collection of characters which describes the knowledge of the system state to the user and is a service which the user can use for further decision making.

IV. EXTENSIBLE WEB – *x*WEB

A. *xWeb Design Methodology*

The *x*Web design methodology provides three levels of abstraction namely; (1) conceptual, (2) logical or schema and (3) document levels. Intuitively, *x*Web methodology provides a 3-step design process to engineer web contents. At the conceptual level we; (1) design the web site and its semantics (such as site layout, structure, data, user access control) using a generic UML model (shown in Fig. 5), which serves as the XML repository for the web site, (2) develop abstract user interface definitions [42] using abstract (web) user interface (AUI) objects or user perspectives [14] and (3) we derive

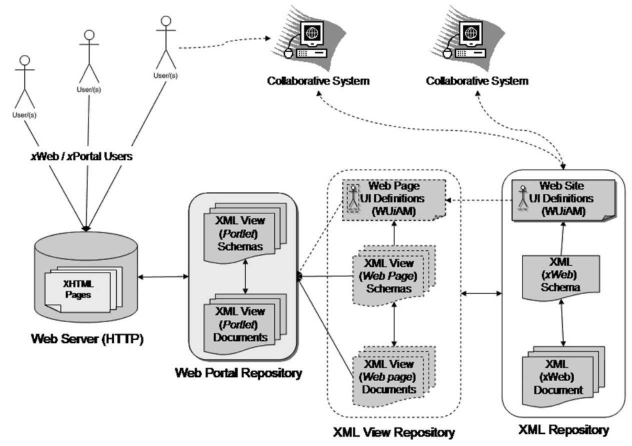


Fig. 5. xWeb (and xPortal) context diagram.

conceptual views to build the web pages (i.e. a web page construct corresponds to one (or more) conceptual view/(s)) and web portals (view of a view/ aggregate view).

At the logical level we; (1) transform the XML repository captured in UML model to XML (schema and document) using the website data (page contents, layouts, resources etc) as shown in Fig. 5, (2) transform AUI objects to web user interface (WUI) definitions such as stylesheet definitions and (3) transform conceptual views to XML view schema definitions (schema & constructor or XQuery [54] definitions) using transformation rules described in [45][49][50]. And finally at the document level, the transformation is threefold; (1) fill XML repository with web data, (2) transform XML view definitions to (imaginary) XML (view) documents (materialized views) and (3) generate XHTML documents using materialized XML (view) documents and UI definitions (an XSLT transformation).

B. xWeb Model & Components

In this section, we briefly discuss the major components of the *x*Web (Fig. 5), its purpose and the implementation model of the components. *x*Web composes of three logical components namely the XML repository, XML view repository and the *x*Web page server. These individual components are discussed in the following sections.

1) XML (Web) Repository: The *x*Web repository hosts the website contents (site meta-data, data and user interface definitions) in a generic XML encoded textual format. The repository composes of a descriptive, semantically rich repository (XML) schema and an XML document that store the web contents. Therefore the XML repository serves as organized web information source for building, hosting and distributing web data for intend use. Simply said, it replicates a simple yet generic XML based database management system (DBMS) hosting web contents.

Storing and maintaining web content in such an XML repository reduces structural ambiguity among web content, yet maintain semantic richness of each individual web object (this in contrast to hosting web content as relational data). Some of the perceived benefit of such repository include and not limited to; (1) generic, yet semantically descriptive web content repository (a direct result of using XML and XML Schema for content description, (2) semantically richer than relational and/or HTML counter parts where schema descriptions are limited and/or optional, (3) since it is in native XML format, the web content dissemination (such as publication data in our motivating example website) among collaborators and/or stakeholders are easier than using propriety messaging formats, (4) based on native XML technology, thus data is descriptive and support heterogeneous web structures and (5) since the web content is independent from WUI objects and/or constraints (in contrast to HTML data), the data along with the WUI definitions (not technology specific presentation layers) are readily distributable and re-usable in other applications such as web portal generation and collaboration. In the case of collaborative website engineering, using XML repository helps in; (1) keeping the content generic yet semantically descriptive, (2) XML (view) Schema driven, therefore no need of additional schema mapping at the source and the target in the content distribution chain, (3) text based data (XML) data thus support for Unicode & multilingual support and (4) keeping the captured web user interface (WUI) definitions independent and free of the web content structure and format.

For physical storage model for the repository, it is readily implemented and hosted in any data storage technology that provides support for native XML documents (and schema) manipulations (from XML-enabled high end DB servers such industrial relational or simple custom made native XML DBs).

2) XML (View) Repository: The XML (view) repository is only logically (not physically) different from its web repository counterpart. Here, instead hosting the original web content and its associated structures, it hosts maintain (XML) views. An XML view represents a web "user" screen (with web contestants and associated WUI definitions) in XML. The description and the semantics of the user screen is provides by the XML view Schema, which in addition provide validity the user screen. Typically, the XML views in the repository organized in a hierarchical manner that loosely reflect a web document tree structure (such root view/node closely resembles the classical index.html page in a classical web server documents and so on).

Physically, the view repository is part of the XML (web) repository storage model (similar to that of external schema in the relational databases) and implemented as part of the XML (web) repository (external schema). Typically the XML views and the schemas are persistence in the view repository together with their view definitions as they serve as the middleware (thus avoiding overhead of additional XML-aware middleware) between the presentation oriented XHTML pages and the core XML repository based web contents. Here, the middleware support is provided in the form of (XML) view updates and re-writes.

3) xWeb (XHTML) page Sever: The *x*Web Page server is a typically a web server serving clients of X/HTML pages. The XHTML pages are generated (preferably not in real-time) using the XML views stored in the XML (view) repository (in batch mode, depending on the web content type) and the WUI definitions using XSLT transformation. A detail discussion of such transformation (XML view to XHTML) can be found in [38] in the context of web portals. The main advantage of using XHTML based pages to build screens is performance and compatibility. Other perceived advances of using HTML based screens include; (1) no propriety standards (classical HTML server pages) or browsers needed to view the pages (though originally the web content is based in an XML encoded format), (2) easy to implement and maintain (server/client technologies), (3) no new scripts and/or web languages embedded into the web pages and (4) no middleware and/or servers are needed.

V. A WALKTHROUGH OF THE *x*WEB DESIGN METHODOLOGY

In this section, we provide a practical walk though of the *x*Web methodology using a case study example. To understand *x*Web, it is important to understand both the inter-related *x*Web architecture and the design methodology itself. The design-time methodology has eight steps to model and deploy a website. But first we briefly describe the case study.

A. An Illustrative Case Study

To demonstrate our work, we developed our research group's website eXel [55] as a simple *x*Web/ *x*Portal [22][38] system. It is a simple web site used as information source (research collaboration, news, publication list etc.), for public relations (PR), references (for members and their students/collaborators) and for collaborative work (with other research and industry partners/entities). In its simple form it has four user groups (1) visitors (any user/agent visiting site), (2) collaborative partners, (3) members and (4) administrators. Each group has some predefined privileges in regards to accessing the web site visitor being the least privileged group and the administrator (here we assume that administrator is a member of the research group) being the highest privileged group. Each group use a predefined web portal (and associated portlets), namely; (1) open/generic web portal, (2) member-portal (3) collaborator-portal and (4) admin-portal. A Use-case model of this is shown in Fig. 6.

In related work [56][57], we discussed in detail about communities (open, closed and locked) based on access privileges for distributed information systems such as web portals. Based on that, here in the xPortal system, member and administrator group belongs to locked community whilst collaborative partner group belongs to closed community. The visitors (and other web users) belong to open community where they mainly have view privileges only. The open-portal is the typical web view for all visitors of the site. Its main use is to display public information without access to add, edit or delete permissions. The collaborator portal is for approved collaborative partners of the eXel site, where they have some added privileges (update publication list, download selected publications etc) than visitors. Each member of the research group has their own portal and is allowed to view, edit and delete their own work.

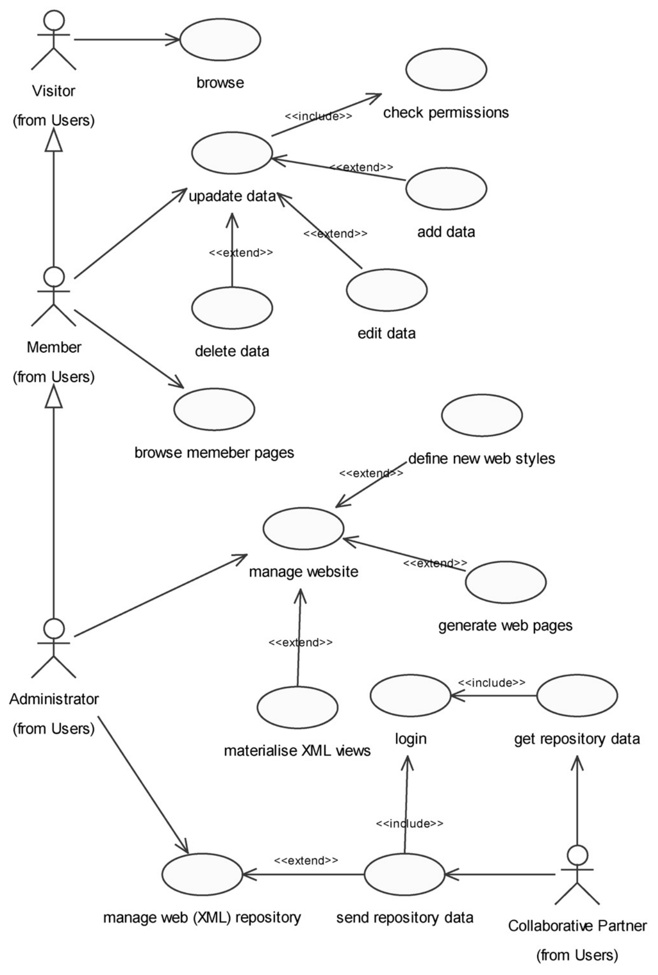


Fig. 6. Use-Case model of the example case study

Step 1. Development of conceptual model of the web domain in question: At the conceptual level, the website domain and data requirements are analysed and captured (using UML) using typical OO conceptual modelling techniques with extensive use-case analysis (discussed in Section 5.2) and WUiAM analysis (discussed in Section 5.3). Also identified are the abstract (web) user interface (AUI) classes [14], such as screens, their components and the navigation links.

B. Use-Case Analysis

Though we do not intend to address the issue of user access control design for web sites, in regards to privilege and access control in *x*Web based systems, we look back at work done in [56][57], where we discussed in detail about communities (open, closed and locked) for web based systems based on their access privileges. Based on that, in our case study example, we identified four main groups of actors who interact with system. They are (1) members, (2) administrators, (3) collaborators and (4) visitors. The member and administrator group belongs to the locked community while collaborative partner group belong to closed community. The visitors (and other web users) belong to open community where they mainly

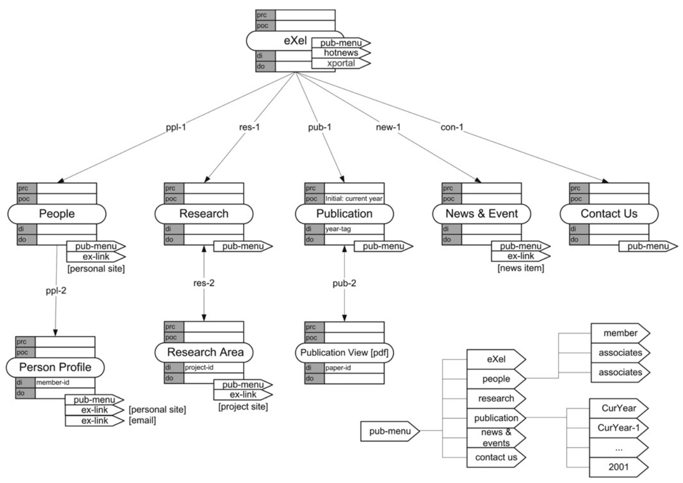


Fig. 7. CNN model of the public portion of the eXel web site.

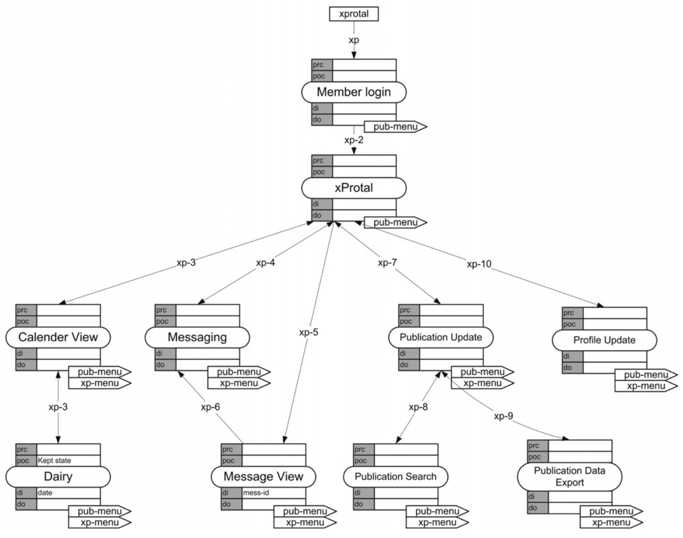


Fig. 8. CNN model of the xPortal portion of the eXel web site.

have view privilege only. The complete use-case analyses together with use-cases are shown in Fig. 6.

C. WUiAM Analysis

In this section, two of the CNN model examples are shown here. This includes the model of the open and portal section of the eXel web site, shown in Fig. 7 and Fig. 8 respectively. Note here that the purpose of CNN analysis model are to identify the Abstract Web User Interface classes construct, which include the high level information and command components and navigational attribute. The most important notion of CNN is to capture the possible navigation path of user based on user analysis, and therefore the necessary navigational feature that the system must provides, and this include the consideration of pre-condition, and post-condition of possible transaction of user interaction. In the model, the basic structure of the system is depicted, based on this information; the navigational structure can also be defined. The public section of the eXel web site are mainly consist of a context that are compose with static information, such as member's detail,

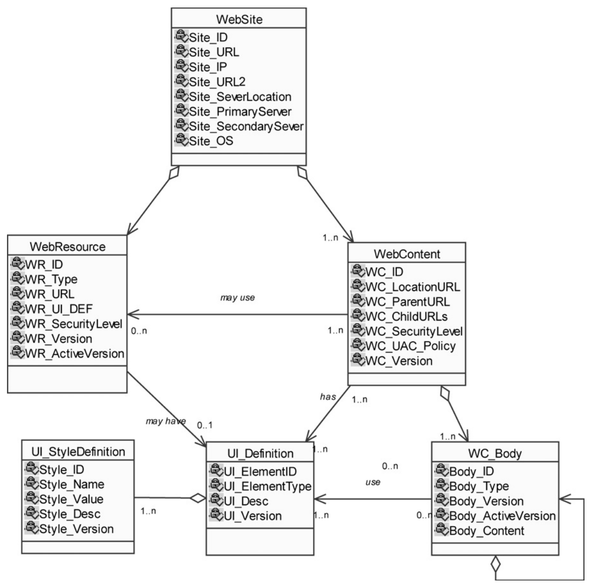


Fig. 9. XML (web) Repository (conceptual level).

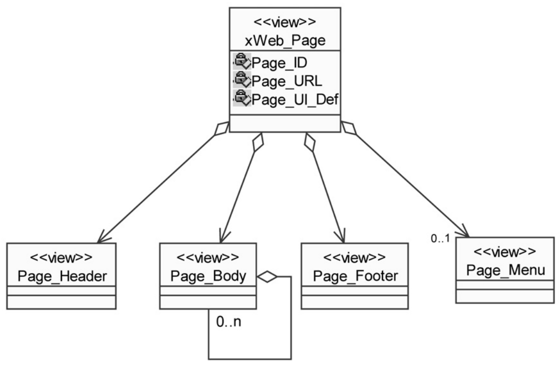


Fig. 10. XML (view) Repository (conceptual level).

contact information, etc. The only semi-dynamic content is in the Publication context, where member's publications are listed and some publications may be linked to the actual file that can be view or download by visitors. Note here that all external navigational requirements are also captured by the model, which we argue are important element for the establishment of the completeness of a user level model for any web development.

The WUiAM model here formed part of the overall XML (web) repository, its relationship with the actual data from the Layered View Model is maintained relaying on the XML view schema on this level. The details captured in the WUiAM and the conceptual model of the XML (web) repository from the core part of conceptual layer in the Layered View Model of the *x*Web methodology, and provide the foundation for the transformation of the model into low level design model of the overall system.

Step 2. Development of conceptual model of the XML (web) repository: Based on the conceptual model development in Step 1, the conceptual model is mapped to the XML (web) repository model under three categories namely; (1) web contents, (2) web resources (hypermedia objects host at the

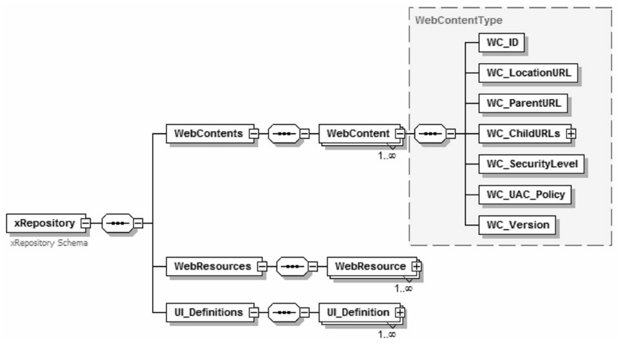


Fig. 11. XML (web) Repository (logical model).

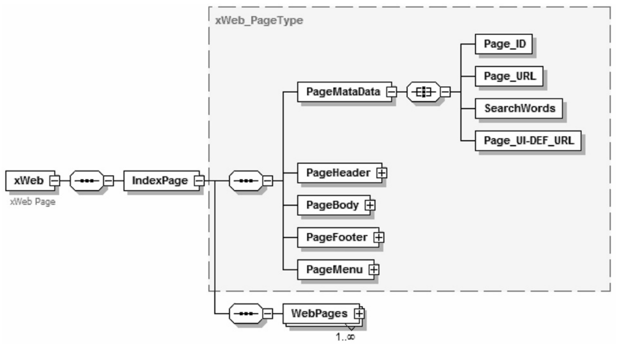
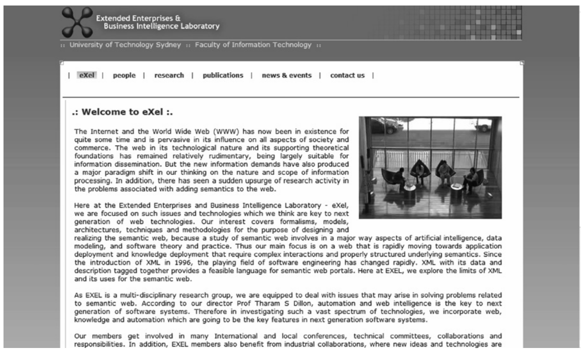


Fig. 12. XML (View) Repository (logical model).


Extended Enterprises & Business Intelligence Laboratory

:: University of Technology Sydney :: Faculty of Information Technology ::

| eXel | people | research | publications | news & events | contact us |

.: Welcome to eXel :.

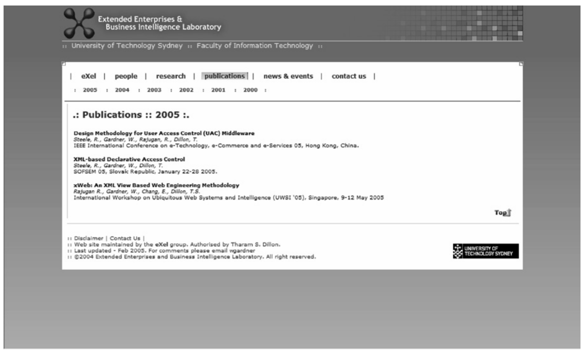
The Internet and the World Wide Web (WWW) has now been in existence for quite some time and is pervasive in its influence on all aspects of society and commerce. The web in its technological nature and its supporting theoretical foundations has remained relatively rudimentary, being largely suitable for information dissemination. But the new information demands have also produced a major paradigm shift in our thinking on the nature and scope of information processing. In addition, there has seen a sudden upsurge of research activity in the problems associated with adding semantics to the web.

Here at the Extended Enterprises and Business Intelligence Laboratory - eXel, we are focused on such issues and technologies which we think are key to next generation of web technologies. Our interest covers formalisms, models, architectures, techniques and methodologies for the purpose of designing and realizing the semantic web, because a study of semantic web involves in a major way aspects of artificial intelligence, data modeling, and software theory and practice. Thus our main focus is on a web that is rapidly moving towards application deployment and knowledge deployment that require complex interactions and properly structured underlying semantics. Since the introduction of XML in 1996, the playing field of software engineering has changed rapidly. XML with its data and description tagged together provides a feasible language for semantic web portals. Here at EXEL, we explore the limits of XML and its uses for the semantic web.

As EXEL is a multi-disciplinary research group, we are equipped to deal with issues that may arise in solving problems related to semantic web. According to our director Prof Tharam S Dillon, automation and web intelligence is the key to next generation of software systems. Therefore in investigating such a vast spectrum of technologies, we incorporate web, knowledge and automation which are going to be the key features in next generation software systems.

Our members get involved in many International and local conferences, technical committees, collaborations and responsibilities. In addition, EXEL members also benefit from industrial collaborations, where new ideas and technologies are

Fig. 13. Typical xWeb page examples (XHTML).


Extended Enterprises & Business Intelligence Laboratory

:: University of Technology Sydney :: Faculty of Information Technology ::

| eXel | people | research | publications | news & events | contact us |

: 2005 : 2004 : 2003 : 2002 : 2001 : 2000 :

.: Publications :: 2005 :.

Design Methodology for User Access Control (UAC) Middleware
Steele, R., Gardner, W., Rajugan, R., Dillon, T.
IEEE International Conference on e-Technology, e-Commerce and e-Services 05, Hong Kong, China.

XML-based Declarative Access Control
Steele, R., Gardner, W., Dillon, T.
SOFSEM 05, Slovak Republic, January 22-28 2005.

xWeb: An XML View Based Web Engineering Methodology
Rajugan R., Gardner, W., Chang, E., Dillon, T.S.
International Workshop on Ubiquitous Web Systems and Intelligence (UWSI '05), Singapore, 9-12 May 2005

Top

:: Disclaimer | Contact Us |
:: Web site maintained by the **eXel** group. Authorised by Tharam S. Dillon.
:: Last updated - Feb 2005. For comments please email wgardner
:: ©2004 Extended Enterprises and Business Intelligence Laboratory. All right reserved.

UNIVERSITY OF TECHNOLOGY SYDNEY

Fig. 14. Typical xWeb Page examples (XHTML).

website)and (3) user interface definitions (such as AUI). The model developed for the example case study is shown in Fig. 9.

Step 3. Development of website conceptual model: Here, based on XML (web) repository, conceptual views (views at the conceptual level) are constructed to satisfy one "screen" which in turn satisfy one or more web user task/(s). The conceptual views constructed form the XML (view) repository. Fig. 10 shows this model, which is developed for the case study example.

Step 4. Development of the XML (web) repository logical model: Here, the conceptual model of the XML (web) repository is mapped to XML schema using schemata transformation rules defined by Ling Feng *et al.* [45][49][50]. This (XML) logical/schema model is shown in Fig. 11.

Step 5. Development of the XML (view) repository logical model: The conceptual views defined in Step 3 are mapped to XML views, where XML view schemas are constructed (an example based on the case study is shown in Fig. 12).

Step 6. Development of the XML (web) repository document model: At this stage, the XML (web) repository is populated with the web contents. Later the populated repository (XML) document/(s) are validated against the XML (web) repository schema generated during Step 4.

Step 7. Development of the XML (view) repository document model: Here, the XML views defined are materialised and validated against the (XML) view schema constructed in Step 5 above. The view materialization can be done using automated tools (which uses XQuery or SQL 2003) or manually using simple XPath or XSLT queries.

Step 8. Development of the *x*Web (XHTML) Pages (Fig. 13-14): Here the materialised XML views are transformed into XHTML pages using XSLT and style-sheet definitions, thus enabling classical web servers to host *x*Web pages. An example of such transformation is discussed in [38].

VI. Conclusion and Future Work

In this paper, we have demonstrated how XML-view coupled with WUiAM can provide a strong data semantics oriented and user centric website design methodology at the same time. It provides an approach to design and implement and manage semantically rich websites manage using MDA like initiatives.

For future work, a few issues deserve investigation. First, the process of automation, where the schemata transformation, view construction and model mapping between the levels (conceptual, logical and document) are automated. Second, is the investigation into XForms as (web) user interface objects. Another area that needs refinement is handling interactive (web) objects (to support web applications), where web contents are updated in the XML (web) repository via XML (view) repository in real-time. Also investigation into formally modeling user access control (UAC) [58] mechanisms as part of the *x*Web methodology.

References

[1] Booch, G. (1993) *Object-oriented analysis and design with applications*. Benjamin/Cummins series in object-oriented software engineering, Redwood City, Calif.

[2] Abiteboul, S. *et al.*(2002) Active XML: A Data-Centric Perspective on Web Services. *BDA.*

[3] Abiteboul, S. *et al.* (1999) Active Views for Electronic Commerce. *Proceedings of the 25th International Conference on VLDB*. Edinburgh, Scotland.

[4] Aragones, A. and Hart-Davidson, W. (2002) Why, When and How do Users Customize Web Portels? *Proceedings of IEEE International Professional Communication Conference (IPCC '02).*
Booch, G. (1993) *Object-oriented analysis and design with applications*, The Benjamin/Cummings series in object-oriented software engineering. Redwood City, Calif. Reading, Mass.: Benjamin/Cummings Pub. Co.; Addison-Wesley.

[5] Dillon, T. S. and Tan, P. L. (1993) *Object-Oriented Conceptual Modeling*: Prentice Hall, Australia.

[6] Conallen, J. (2003) *Building Web Application with UML*: Addison-Wesley.

[7] J. Conallen (1999) Modeling Web Application Architectures with UML. *Communictaions of the ACM*, **42**(10)

[8] Gaedke, M., Segor, C. and Gellersen, H.-W. (2000) WCML: paving the way for reuse in object-oriented Web engineering. *Proceedings of the ACM symposium on Applied computing (SIGAPP '00).* Italy: ACM Press.

[9] Lei, Y., Motta, E. and Domingue, J. (2002) An Ontology-Driven Approach to Web Site Generation and Maintenance. *Proceedings of 13th International Conference on Knowledge Engineering and Management.* Sigenza, Spain: Springer-Verlag.

[10] Lei, Y., Motta, E. and Domingue, J. (2004) OntoWeaver-S: Supporting the Design of Knowledge Portals. *Proceedings 14th International Conference on Knowledge Engineering and Knowledge Management (EKAW '04).* Northamptonshire, UK.

[11] W3C-WS (2002) *Web Services Activity, (http://www.w3.org/2002/ws/).* 2002

[12] Gardner, W., Chang, E. and Dillon, T. S. (2003) Two Layer Web User Interface Analysis Framework Using SNN and iFIN. *On The Move to Meaningful Internet Systems 2003: OTM 2003 Workshops HCI-SWWA.*

[13] Tatnall, A. (2004) *Web Portals: The New Gateways to Internet Information and Services*, Idea Group.

[14] Chang, E. and Dillon, T. S. (1994) Integration of User Interfaces with Application Software and Databases Through the Use of Perspectives. *1st International Conference on Object-Role Modeling (ORM '94).* Australia.

[15] Poon, P. M. S., Dillon, T. S. and Chang, E. (2004) Transformation of QoS data into XML characterizing data communication in Real Time Distributed Systems. *2nd IEEE International Conference on Industrial Informatics (INDIN '04).* Berlin, Germany.

[16] Lei, Y., Motta, E. and Domingue, J. (2004) Modelling Data-Intensive Web Sites with OntoWeaver. *Proceedings of the International Workshop on Web Information Systems Modelling (WISM '04).* Riga, Latvia: Springer-Verlag.

[17] W3C-XML (2004) *Extensible Markup Language (XML) 1.0, (http:/www.w3.org/XML/).* (Web) 2004 4th February 2004.

[18] W3C-XSD (2004) *XML Schema, (http://www.w3.org/XML/Schema).* 2004.

[19] OMG-MDA (2003) *The Architecture of Choice for a Changing World, MDA Guide Version 1.0.1 (http://www.omg.org/mda/).* 2003

[20] OMG-UML (2003) *UML 2.0 Final Adopted Specification (http://www.uml.org/UML2.0).*

[21] Elmasri, R. and Navathe, S. (2004) *Fundamentals of database systems.* New York: Pearson/Addison Wesley.

[22] Rajugan, R. *et al.* (2005) xWeb: An XML View Based Web Engineering Methodology. *International Workshop on Ubiquitous Web Systems and Intelligence (UWSI '05), Colocated with ICCSA '05.* Singapore.

[23] Gardner, W., Chang, E. and Dillon, T. S. (2003) Analysis Model of Web User Interface for Web Applications. *16th International Conference on Software and Systems Engineering and their Applications ICSSEA 03.* France.

[24] Rajugan, R. *et al.* (2005) A Three-Layered XML View Model: A Practical Approach. *24th International Conference on Conceptual Modeling (ER '05).* Klagenfurt, Austria: Springer-Verlag.

[25] Troyer, O. M. F. and Leune, C. J. (1998) WSDM: a User Centered Design Method for Web Sites. *7th International World Wide Web Conference.*

[26] Ceri, S., Fraternali, P. and Bongio, A. (2000) Web Modeling Language (WebML): a Modeling Language for Designing Web Site. *9th International World Wide Web Conference, WWW2000.*

[27] Gant, J. P. and Gant, D. B. (2002) Web portal functionality and State government E-service. *Proc.of the 35th Annual Hawaii International Conference on System Sciences (HICSS '02).* Hawaii.

[28] Reynolds, D., Shabajee, P. and Cayzer, S. (2004) Semantic Information Portals. *Proceedings of the 13th International World Wide Web Conference (WWW '04).* USA.

[29] Bellas, F., Fernández, D. and Muiáo, S. (2004) A Flexible Framework for Engineering "My" Portals. *13th International World Wide Web Conference, WWW2004.*

[30] Aragones, A. and Hart-Davidson, W. (2002) Why, When and How do Users Customize Web Portels? *Proceedings of IEEE International Professional Communication Conference (IPCC '02).*

[31] Gu, A., Henderson-Sellers, B. and Lowe, D. (2002) Web Modeling Languages: the gap between requirements and current exemplars. *8th Australian World Wide Web Conference, AUSWEB'02.*

[32] Garzotto, F., Paolini, P. and Schwabe, D. (1993) HDM – A Model-Based Approach to Hypertext Application Design. *ACM Transactions on Information Systems (TOIS)*, **11**(1): 1–26.

[33] Schwabe, D., Rossi, G. and Barbosa, S. D. J. (1996) Systematic hypermedia application design with OOHDM. *The 7th ACM conference on Hypertext and Hypermedia*: ACM Press.

[34] Chang, E. and Dillon, T. S. (1999) Software Development Methodology and Structure of a Class of Multimedia Web Based Systems. *2nd Asia-Pacific Web Conference - Technologies and Applications for the New Millennium, APWeb 99.* Hong Kong.

[35] Chan, S., Dillon, T. S. and Siu, A. (2002) Applying a mediator architecture employing XML to retailing Inventory Control. *The Journal of Systems and Software*, **60**: 239–248.

[36] Do, H. H. and Rahm, E. (2004) Flexible integration of molecular-biological annotation data: The genmapper approach. *Proceedings of the 9th International Conference on Extending Database Technology (EDBT '04).* Heraklion, Crete, Greece.

[37] Gruber, T. R. (1993) A Translation Approach to Portable Ontologies. *Knowledge Acquisition*, **5**(2): 199–220.

[38] Gardner, W. *et al.* (2004) xPortal: XML View Based Web Portal Design. *17th International Conference on Software and Systems Engineering and their Applications ICSSEA 04.* Paris, France.

[39] OMG-UML (2003) *Unified Modeling Language (UML) Version 1.5 Specification.* formal/03-03-01. 2003, OMG.

[40] Rajugan, R. *et al.* (2005) Semantic Modelling of e-Solutions Using a View Formalism with Conceptual & Logical Extensions. *3rd International IEEE Conference on Industrial Informatics (INDIN '05).* Perth, Australia: IEEE Computer Society.

[41] Rajugan, R. *et al.* (2005) A Layered View Model for XML Repositories & XML Data Warehouses. *The 5th International Conference on Computer and Information Technology (CIT '05).* Shanghai, China: IEEE CS Press.

[42] Chang, E. (1996) *Object Oriented User Interface Design and Usability Evaluation.* Department of Computer Science & Computer Engineering. 1996, La Trobe University, Melbourne, Australia.

[43] Feng, L., Chang, E. and Dillon, T. S. (2002) A Semantic Network-based Design Methodology for XML Documents. *ACM Transactions on Information Systems (TOIS)*, **20, No 4**(4): 390–421.

[44] Feng, L. *et al.* (2003) An XML-Enabled Association Rule Framework. *14th International Conference on Database and Expert Systems Applications (DEXA '03) 2003.* Prague, Czech Republic: Springer-Verlag Heidelberg.

[45] Feng, L., Chang, E. and Dillon, T. S. (2003) Schemata Transformation of Object-Oriented Conceptual Models to XML. *International Journal of Computer Systems Science & Engineering*, **18, No. 1**(1): 45–60.

[46] Conrad, R., Scheffner, D. and Freytag, J. C. (2000) XML conceptual modeling using UML. *19th International Conference on Conceptual Modeling (ER '00).* USA: Springer-Verlag London, UK.

[47] Golfarelli, M., Maio, D. and Rizzi, S. (1998) The Dimensional Fact Model: A Conceptual Model for Data Warehouses. *International Journal of Cooperative Information Systems*, **7**(2–3): 215–247.

[48] Trujillo, J. *et al.* (2001) Designing Data Warehouses with OO Conceptual Models. *Computer*, 66–75.

[49] Xiaou, R. *et al.* (2001) Modeling and Transformation of Object-Oriented Conceptual Models into XML Schema. *12th International Conference on Database and Expert Systems Applications (DEXA '01) 2001*: Springer.

[50] Xiaou, R. *et al.* (2001) Mapping Object Relationships into XML Schema. *Proceedings of OOPSLA Workshop on Objects, XML and Databases.*

[51] Rajugan, R. *et al.* (2005) XML Views, Part III: Modeling XML Conceptual Views Using UML. *7th International Conference on Enterprise Information Systems (ICEIS '05).* Miami, USA.

[52] Wouters, C. *et al.* (2004) A Practical Approach to the Derivation of a Materialized Ontology View. In: D. Taniar and W. Rahayu, Editors., *Web Information Systems,* Idea Group Publishing.

[53] Chang, E. and Dillon, T. S. (1998) The Navigational Aspects of the Logical Design of User Interfaces. *First International Symposium on Object-Oriented Real-Time Distributed Computing, IEEE ISORC '98.*

[54] W3C-XQuery (2004) *XQuery 1.0: An XML Query Language*. XML Query Language (XQuery) (Web) 2004 November 2003.

[55] EXEL-Lab (2004) *Extended Enterprises & Business Intelligent Laboratory, (http://exel.it.uts.edu.au/).* 2004

[56] Chang, E. *et al.* (2001) Virtual Collaborative Logistics and B2B e-Commerce. *e-Business Conference*. Duxon Wellington, NZ.

[57] Chang, E. *et al.* (2003) A Virtual Logistics Network and an e-Hub as a Competitive Approach for Small to Medium Size Companies. *2nd International Human.Society@Internet Conference*. Seoul, Korea.

[58] Steele, R. *et al.* (2005) A Design Methodology for User Access Control (UAC) Middleware. *Proceedings of the IEEE International Conference on e-Technology, e-Commerce and e-Service (EEE '05).*

Rajugan Rajagopalapillai holds a bachelor's degree in Information Systems from La Trobe University, Australia. He has worked in the industry as chief application/database programmer in developing sports planing and sports fitness & injury management software and as database administrator. He was also involved in developing an e-Commerce solution for a global logistics (logistics, cold-storage and warehousing) company as a software engineer/architect. He is currently a PhD student at University of Technology, Sydney (UTS) Australia. He has published research articles which have appeared in international refereed conference and journal proceedings. His research interests include Object-Oriented conceptual models, XML, data warehousing, software engineering, database and e-commerce systems. He is a member of IEEE, member of ACM and an Associate member of Australian Computer Society (AACS).

William Gardner is a Ph.D. candidate at University of Technology, Sydney, Australia. His research interests lies in web engineering. In particular he focus on web user interface design, object oriented modelling, model driven development, and XML technologies, eHealth and eLogistics system development. He has worked in a number of industry projects as application programmer, and software architect in developing of usability metric software. He has also led the development of a large scale e-Commerce/e-Business supply chain solution for a global logistics company. He has published research articles which have appeared in many international refereed conferences and journals.

Professor Elizabeth Chang is the Director of Area Research Excellence for Frontier Technology and the Centre Extended Enterprise and Business Intelligence (CEEBI) and Professor of IT in School of Information Systems. She has over 100 scientific conference and journal papers in IT and numerous invited Keynote papers at International Conferences. All her research and development work are in the area of IT Applications, Software Engineering and Logistics Informatics. Her research interests include issues related to the process of producing an IT application, methodologies, Software and System Architecture, web services or Mediator based systems, Peer to Peer Communications, Trust Management, XML, XQL, RDF and Ontology, Mobile agents, Security and Privacy, Reliability and Fault Tolerance, Usability Metrics and application areas such as Logistic Informatics and Bio-informatics.

Professor Tharam S. Dillon is the Dean of the Faculty of Information Technology at University of Technology, Sydney (UTS) in Australia. His research interests include data mining, internet computing, e-commerce, hybrid neuro-symbolic systems, neural nets, software engineering, database systems and computer networks. He has also worked with industry and commerce in developing systems in telecommunications, health care systems, e-commerce, logistics, power systems, and banking and finance. He is editor-in-chief of the *International Journal of Computer Systems Science and Engineering* and the *International Journal of Engineering Intelligent Systems*, as well as co-editor of the *Journal of Electric Power and Energy Systems*. He is on the advisory editorial board of *Applied Intelligence* published by Kluwer in the US and *Computer Communications* published by Elsevier in the UK. He has published more than 400 papers in international and national journals and conferences and has written four books and edited five other books. He is a fellow of the IEEE, fellow of the Institution of Engineers (Australia), and fellow of the Australian Computer Society.