# WebML+ : a Web modeling language for forming a bridge between business modeling and information modeling

Rachatrin Tongrungrojana, David Lowe
University of Technology, Sydney
PO Box 123 Broadway NSW 2007, Australia
email: {rachatrin.tongrungrojana, david.lowe}@uts.edu.au

## Abstract

*One aspect of the development of Web-enabled systems that has received increasing attention is information modeling, particularly with respect to aspects such as navigation and content models. These models have however typically focused on modeling at a relatively low-level and have failed to address higher-level aspects, such as architectural and business process modeling. We have proposed WebML+, a set of formal extensions to an existing modeling language (WebML), which is able to form a bridge between higher level business models and lower level detailed designs. In this paper we provide guidelines for supporting the derivation of a WebML+ model from/to these different models, thereby providing a concrete link between them. We illustrate these guidelines with a detailed example.*

## 1 Introduction

In the span of a decade, the Web has transformed entire industries and entered the mass culture. It has created new expectations on ease of access, freshness, and relevance of information accessed via the Internet from commercial, government, and academic Web systems. Web-enabled systems are becoming increasingly crucial to almost all sectors in society [18]. This has been accompanied by a rapid increase in the complexity of these Web systems.

Various approaches have been developed for representing this complexity. At lower levels there are detailed design models that cover both functional and informational aspects. However these approaches still have various limitations. For instance, all current approaches lack the ability to model Web systems at higher levels of abstraction and to link effectively with business models and processes. Work on information architectures [16] address these issue to a limited extent, though the notations used are rarely consistent with those used for lower level information modeling, and as a result these are rarely integrated effectively. Similarly, whilst information architectures often support understanding user interactions and engagement with a site [11] and the way in which this influences the information organisation, the nature of the information exchange and the internal inter-relationships between these information domains is often overlooked. Nor do these models usually provide an effective consideration of the information environment in which the system exists.

We have proposed extensions to an existing modeling language (WebML) that address these limitations. The details of this modeling language, referred to as WebML+, are provided elsewhere [19]. A key point of these extensions is that the WebML+ approach is built around the notion of information flows at the level of connecting to business processes. This enables the models to form a link between higher level business models and lower level design models – a characteristic that is crucial in Web-development, where the systems under development often lead to fundamental changes in business processes and models. This implies the ability to support the derivation of other level models from/to the WebML+ model. In this paper we focus on the presentation of WebML+ concepts in linking higher level business models to lower level detailed design models. In Section 2 we consider related work. Section 3 provides a brief overview of the concepts of WebML+ through an illustrative example. Section 4 introduces the ability of WebML+ in linking business models to detailed designs, and then illustrates some examples of the derivations of models. Finally in section 5 we discuss the implications of this research and present some ideas for further work.

## 2 Background

As discussed above, over the last decade we have seen the rapid emergence of systems that utilise web technologies to support the integration of complex functionality with rich information handling. This emergence has been accom-

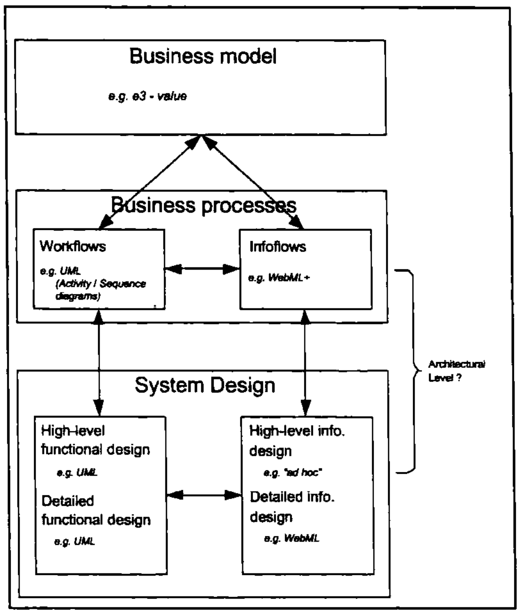

**Figure 1. Typical levels in modelling of web systems**

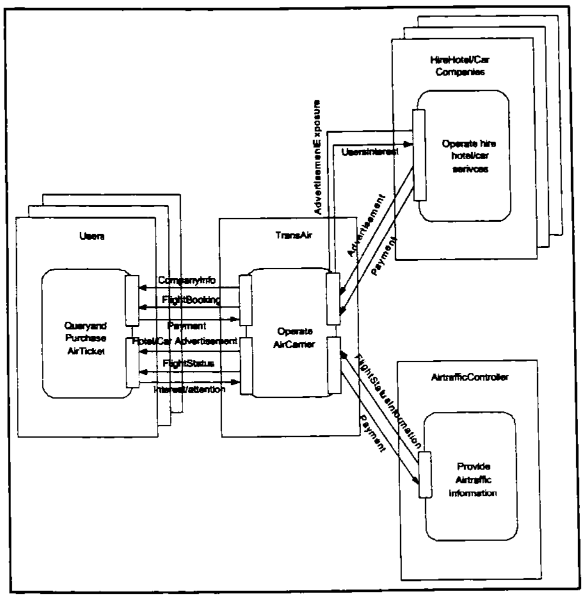

**Figure 2. Typical Business Model represented using $e^3$-value**

panied by the development of modelling languages capable of capturing some – though not all – of the aspects of these systems. To model these system there are a number of elements that we would like to represent.

Figure 1 illustrates several different levels of modelling that might occur in representing the design of web-enabled systems. At the top level we have models of the actual business that is utilising these systems. These models typically represent the value exchanges between the organisation and other entities that enable the organisation to achieve its business goals. Whilst modelling notations at this level are quite diverse and often ad hoc (or at least relatively informal – for example it is common to simply use natural language or simple flow-charting) some more formal approaches do exist. A typical example is the $e^3$-value™ business modelling notation [9, 10]. This model focuses on the core concept of value, and expresses how value is created, interpreted and exchanged within a multi-party stakeholder network. Figure 2 provides an example $e^3$-value business model for a TransAir organisation (described in Section 3). This figure captures the key value exchanges that occur between TransAir and its stakeholders.

Conversely, at the lowest level of Figure 1 we have models of the system design. Functional design models are relatively well-established both within the software literature and within current commercial practice. The dominant model within conventional software development is (arguably) now UML [3, 15]. This can be used to model both detailed design and higher-level designs through a complex suite of diagrams that are all treated as views onto a consistent underlying model. The relationships to business models are however not particular clear [6].

In terms of modelling the information design, the situation is somewhat less mature. Typically we wish to model not only the information itself, but also the relationship between the underlying content and the user-perceived views of that content, the interactions with those views (such as navigational aspects), and the ways in which the information is represented and presented to the users. This modelling tends to be much more complex than traditional "data modelling" (which used approaches such as entity-relationship and data flow modelling). Whilst existing modelling languages (such as UML) can be used to represent the functional aspects, they are not as effective at representing these informational aspects. Although some attempts have been made to adapt UML to support information models (e.g. one interesting work is WAE by Conallen [5]), these are still relatively simplistic and suffer from a notational confusion. For example, modelling constructs normally associated with functional elements are used to represent information aspects, or modelling constructs (such as stereotypes in UML) are used incorrectly or inconsistently and some attempts (such as WAE) tend to focus only on modelling lower level constructs.

A number of modelling approaches that are specific to information design have appeared over the last decade. The earliest of these emerged out of the hypertext community, and include approaches such as RMM [13], OOHDM [17] and others [7, 14]. More recently a number of approaches have been developed that utilise and adapt software modelling approaches, and in particular UML [1, 2, 5, 12]. Of particular interest in this paper, as an exemplar of these modelling approaches, is WebML [4]. WebML incorporates both an XML-based formal description of a model as well as a graphical notation. Whilst the authors of WebML (and many other of the current approaches) claim to address the full development spectrum – including higher-level descriptions of the informational elements of websites, we contend that their focus is primarily on lower level information modelling. These models typically incorporate modelling of the underlying data (in WebML this is referred to as the structural model), the way in which this data is composed into pages (in WebML, the compositional model) and the topology of the page inter-relationships (in WebML, the navigational model). On top of these can be layered presentational and adaptation models.

Approaches such as WebML are effective at modelling lower level constructs, but they do not adequately address higher level architectural issues. Nor do they provide a clear linkage with business process modelling. To elaborate, whilst there has been considerable recent attention on the area of information architectures the notations and models used to represent these information architectures are rarely consistent with those used for lower level information modelling, and as a result these are rarely integrated effectively (for more information, see [16] or `http://argus-acia.com/ia_guide/index.html`).

Similarly, whilst information architectures often address the development of an understanding of user interactions and engagement with a site [11] and the way in which this influences the information organisation, this is often only modelled at the level of information needs and usage scenarios – adopting modelling techniques such as use cases [8]. The nature of the information exchange and the internal inter-relationships between these information domains is often overlooked or nor modelled explicitly. Nor do the low-level information models and the information architectures usually provide an effective consideration of the information environment in which the system exists.

So this raises the question of what information we really should be modelling? We believe that in the same way that functional elements of business processes can be captured in sequence diagrams or work flows, the informational elements can be captured in information flows. These flows should represent the information exchanges that occur within and between the system, the organisation and users, and the relationships between these information "units". This is analogous to the data flow modelling in conventional structured software design except that we are interested in information rather than data – and hence the context used to understand the information is important.

Whilst WebML can provide a good foundation for this modelling, it needs to be extended to capture and represent these higher-level constructs. This includes the external information context and domain, the source and sinks of the information being designed, and the way in which users interact with this information. In this paper we look at how the extensions to WebML+ that we have proposed provide a bridge between the higher level models (business models) and lower level models (detailed design models).

## 3 WebML+

WebML+ enables developers to express the core features of a system at a higher level of abstraction (when compared to normal Web design notations) without committing to detailed architectural designs. It can be considered as an extension to WebML. The purpose of WebML+ modeling is to define both the internal and the external information flows within an organisation at the business process level. As with WebML, we have defined both a graphical notation and an XML-based formal notation for representing WebML+ models (though we do not show the formal XML DTD here – this is described in [19]). The graphical notation is designed to allow it to be effectively communicated to non-technical members of development teams.

Figure 3 shows an example of a WebML+ model for a hypothetical example: TransAir is a fully-fledged airline which takes an innovative look at lowering airfares. The strategy adopted by TransAir is to lower operating costs (such as ticket processing and maintenance of physical office spaces) by running their business activity through an online Web system. The TransAir system allows customers to purchase tickets and enquire for flight and booking information through a Web interface. In this example we have 4 participating actors: the TransAir organisation itself as an internal actor, and three external actors, users, air traffic control organisation, and hire hotel/car companies.In brief, Internal actors such as TransAir provide information directly to the system – in this case flight formation, price information, a set of user policies, a set of frequent flyer policies and company information. The actor TransAir also has financial information exchanges between its organisation and external actors (e.g. invoice and receipt information). With the external actors, the air traffic control organisation provides flight status (e.g. flight arrival times) to the system and exchange financial information with TransAir. Hire Hotel/car companies provide advertisements about their services to TransAir to be included in information provided to users. Users provide queries for information, payment information as well as user information (such as personal information) to the system while they receive user information and profiles, quotation, booking status and flight status from the system.

So, let us consider what is represented in the model. The organisation boundary (shown as a dashed geometrical polygon) encloses a set of information units. These units represent coherent and cohesive domains of information that are managed or utilised by the organisation. All information within a single unit shares a common context and a common derivation (this is explained shortly). They do not map directly to pages or sets of pages – a single web page may contain partial information from multiple units. Similarly, an information unit may be distributed over multiple pages. Different types of information units are represented graphically using different types of icons.

Some information units are provided directly by actors. For example flight status is provided by Air Traffic Controllers, and frequent flyer policies are provided by TransAir. However, many information units are derived from other units rather than being provided explicitly. These derivations (shown as triangles with incoming and outgoing arrows) capture the inter-relationships between the information units. For example, QuotationInfo is derived from the flight information database, the price database, user policies, user profile information and query information provided by the user.

Furthermore, the system boundary (shown as a dotted geometrical polygon) encloses only the set of information units that are utilised and/or managed by the system under consideration. (i.e. The elements enclosed by the system boundary are a subset of the elements enclosed by the organisation boundary. The organisation boundary captures the interface between the organisation and external stakeholders (i.e. it is crucial in supporting the business modeling). Conversely, the system boundary captures those elements of the organisation that can be managed by the system (i.e. it is crucial in supporting the detailed system design).

## 4 Linking business models to WebML+

A major aspect of WebML+ is the ability to define the information flows within an organisation, system and its context (and hence appropriate elements of the information architecture). Further, this model also is able to form a bridge between the business modelling and the lower level information modelling. This implies the ability to support the derivation of a suitable WebML+ model from/to the business model, and an appropriate WebML detailed design model from/to the WebML+ model. As discussed above, modelling notations at the business model level are quite diverse and often ad hoc, with a typical example being $e^3$-value. Whilst an $e^3$-value model focuses on the exchange of value, the WebML+ approach is constructed around the concept of information flows. This means that information exchanges in WebML+ can be related to value exchanges in $e^3$-value, resulting in an improved linkage between business models and WebML+. Using the following sequence of steps we can convert the business model shown in Figure 2 into a WebML+ model that captures architectural-level information flows (shown in Figure 3).

1. *Identify actors:* Add an organisation boundary to the new WebML+ model (shown as a dashed line in Figure 3), then consider each actor and each market segment in the $e^3$-value business model. If the actor represents either the organisation itself or an entity inside the organisation, then create a corresponding actor in the WebML+ model inside the organisation boundary. If the actor is external to the organisation, then add a corresponding actor to the WebML+ model outside the boundary. For example, the actor TransAir in Figure 2 results in the actor TransAir in Figure 3.

2. *Identify initial information units:* Consider each value exchange in the $e^3$-value business model that includes at least one participating actor that is part of the organisation. For each of these value exchanges, identify the exchange of information that must occur to support the value exchange, and then create corresponding information unit(s) in the WebML+ model. If this information is exchanged with an external actor then place the information unit on the organisation boundary, otherwise place it inside the boundary. For example, consider the way the payment from the external companies to TransAir has been used to derive a receipt information unit. Conversely, the Advertisement exposure (which is the value gained by the external company) does not result in any information transfer.

3. *Identify external information flows:* For each information unit on the organisation boundary add an information flow (either incoming or outgoing) to the relevant external actor that supplies or uses that information.

4. *Identify initial internal information flows:* For internal information units and outgoing information units on the organisation boundary consider whether the information can be supplied directly by an internal actor. If so, then add an information flow from the actor to the information unit. For internal information units and incoming information units on the organisation boundary consider whether the information can be utilised directly by an internal actor. If so, then add an information flow from the information unit to the actor.

5. *Identify information unit types:* Consider each information unit in the model. If it is supplied directly from

an actor then define the unit as supplied, otherwise define it as derived. Consider the permanence of the information and identify the information unit as either transient or persistent.

6. *Define information unit derivations:* For each derived information unit in the model, add an information derivation with relevant information flows showing the source of the derivation. Where necessary add additional intermediate internal information units to support the derivation process. *[Note: In some respects this is the most complex step, since it involves the internal design of the information flows within the organisation — and most of this will not be directly derivable from the higher level business model].*

7. *Refine temporal sequencing:* Check each derivation process. Where an information unit is being derived from a transient information source then determine whether an intermediate persistent information unit is required. For example, in Figure 3 the DisplayedFlightStatus information unit is, in part, derived from the supplied Advertisements. The DisplayedFlightStatus information however will typically be provided at a different time to the provision of the Advertisement (which is transient), and so we have had to add an internal AdvertisementDB information unit to provide persistence of the Advertisements.

8. *Identify System boundary:* Consider the WebML+ model and identify those information units and derivations that will be managed by the system (as distinct from being managed by other mechanisms within the organisation – such as other systems or by manual handling). Add a system boundary to the WebML+ model that encompasses the relevant units and derivations.

9. Review the model and refine.

Conversely, business models (using $e^3$-value) can be derived from WebML+ models. More usefully, it is possible to modify business models to reflect changes that are mode to architectural level system models. The following process can be adopted to generate or modify a $e^3$-value e-business model from a WebML+ model.

1. *Identify actors:* Consider each actor in the WebML+ model. For each external actor, add an equivalent actor (or market segment) to the $e^3$-value model. Add a single actor to the $e^3$-value model to represent the composite of all internal actors.

2. *Identify value exchanges:* Consider each information unit on the organisation boundary in the WebML+ model. In each case, identify the value exchange that is supported (and/or modified) by that information unit and add or modify the appropriate value exchange to the $e^3$-value model. Note that the direction of the value exchange may not be the same as the direction of information flow.

3. *Define value ports and value interfaces:* For each value exchange that is defined in step 2, define an appropriate value port and interface.

4. Review the model and refine.

As an example, consider the situation where we modify the information derivations (shown in Figure 3) so that the DisplayedFlightStatus was no longer derived from the avdertisements (i.e. this information did not contain advertisements) but instead the QuotationInfo derivation included Advertisements. We can readily identify the value exchanges that will be affected in the $e^3$-value model. In effect, the Interest/Attention value exchange will be diminished (and in particular, will not provide the type of attention that will be useful to Advertisers, thereby affecting the AdvertisementExposure value exchange, and ultimately the payment than can be sought from Advertisers.

## 5 Linking WebML+ and WebML

We can also define the link between the WebML+ and WebML models. In WebML the structural schema defines the data domain for the application. The hypertext site view model contains two elements: the composition and navigation models. Together these define the abstract information interface. The following process can be adopted to generate a WebML structural schema from a WebML+ model.

1. *Identify initial entities:* For each persistent information unit (both supplied and derived) define an entity in the WebML structural schema. *[rationale: the persistent information units represent domains of information that are likely to translate into content that underpins the Web system.]*

2. *Identify additional entities from derivations:* For each derivation unit in the WebML+ model, identify all persistent information units that act as sources of information for the derivation and create a relationship between the WebML entities associated with these units. For example, the CreateQuotationInfo derivation in Figure 3 implies the need for a relationship between the entities defined based on the UserInfoandprofileDatabase, the UserPolicy, the PriceDatabase and the FlightDatabase information units. *[rationale: for an information unit C to be derived from units A and B, there must be a relationship between A and B (or between these and a third entity)].*

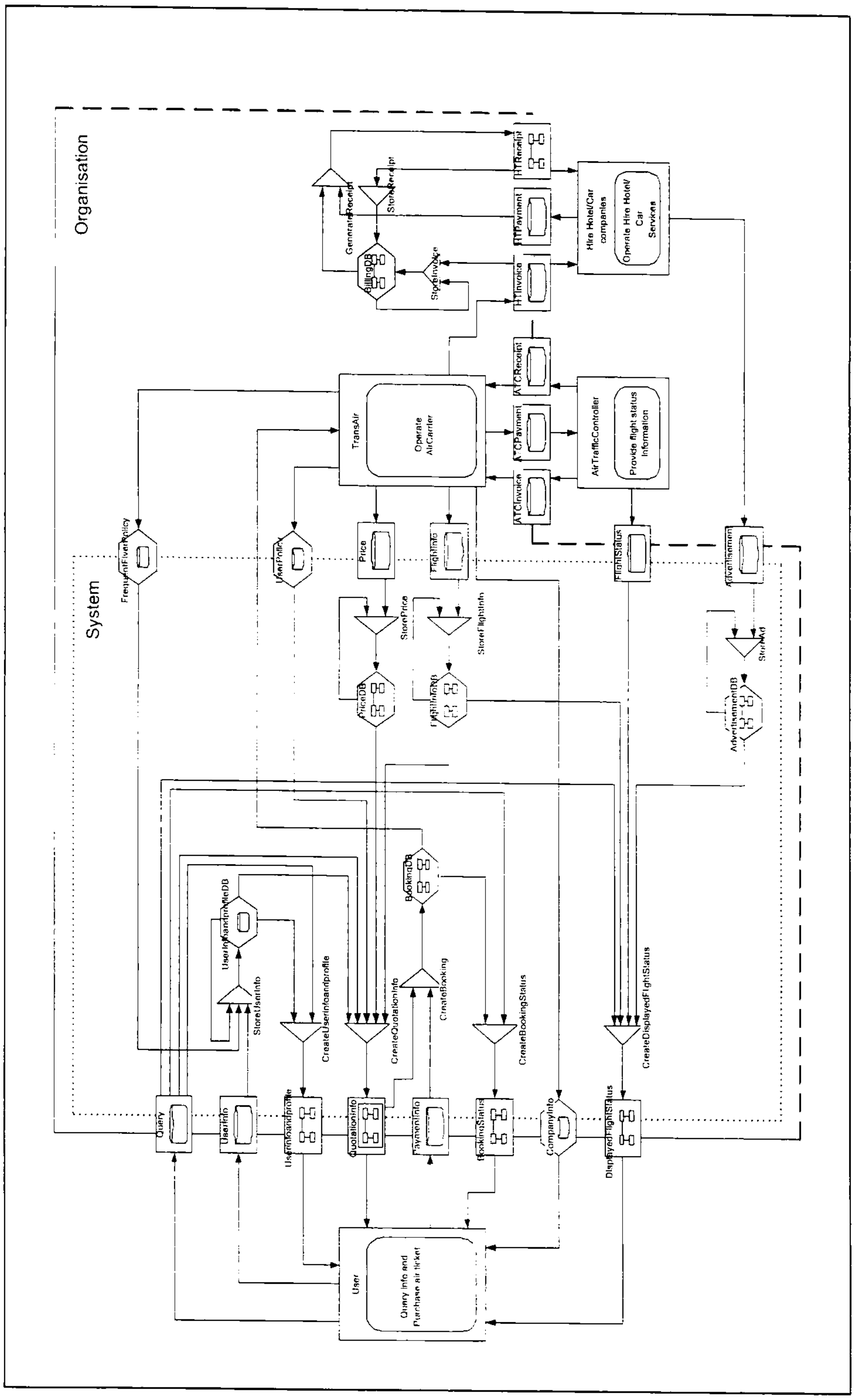

Figure 3. Typical Web System represented using WebML+

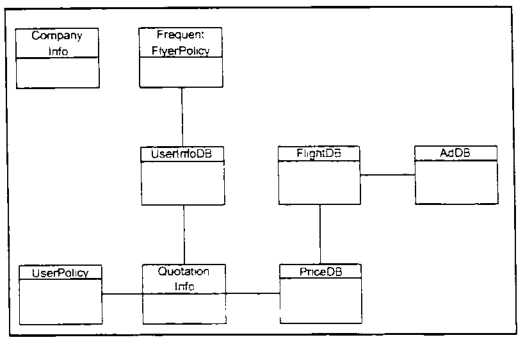

**Figure 4. Example WebML structural schema derived from a given WebML+ model**

3. Where a relationship from step 2 has more than two participating entities introduce an intermediary entity and replace the original relationship with a sequence of binary relationships between the intermediary entity and each of the original entities. *[rationale: WebML only supports binary relationships].*

4. Review the model and refine.

An illustration of the outcome of applying this process to the WebML+ model shown in Figure 3 is given in Figure 4. The process then continues into the definition of the WebML site view. The information units that exist on the system boundary provide a basis for identifying the information with which the external actors interact – and hence the web pages which users are likely to see and navigate between. The following process can be adopted to generate a site view.

1. *Identify areas:* For each information unit in the WebML+ model which has an outgoing information flow directly to an external actor (i.e. it appears as an outgoing information unit that is located on the system boundary) define an area in the WebML site view. For example, the DisplayedFlightStatus info unit in Figure 5 results in the FlighStatus page in Figure 5. (Note: the page in the WebML site view is defined later). *[rationale: the outgoing information unit that is located on the boundary represent domains of information that are available to users.]*

2. *Define units representing information in areas:* Consider each area that is defined in step 1, and then define a data or multi data unit in the WebML site view representing the information unit in the WebML+ model. For example, the CompanyInfo information unit in Figure 3 results in the CompanyInfo data unit in Figure 5. In addition, if the information unit that is regarded in step 1 is a derived information unit, then propose a data or multi data unit in the WebML site view representing a relevant source of information for the derivation in the WebML+ model. (Note: this is optional and might not be suitable for some situations). For example, the FlightStatus area in Figure 5 contains the Query data unit which is resulted from the Query information unit in Figure 3. *[rationale: the supplied information units represent domains of information that are provide directly from information supplier to information requester without derivation/integration process. The derived information units are generated from derivation process from other information units]*

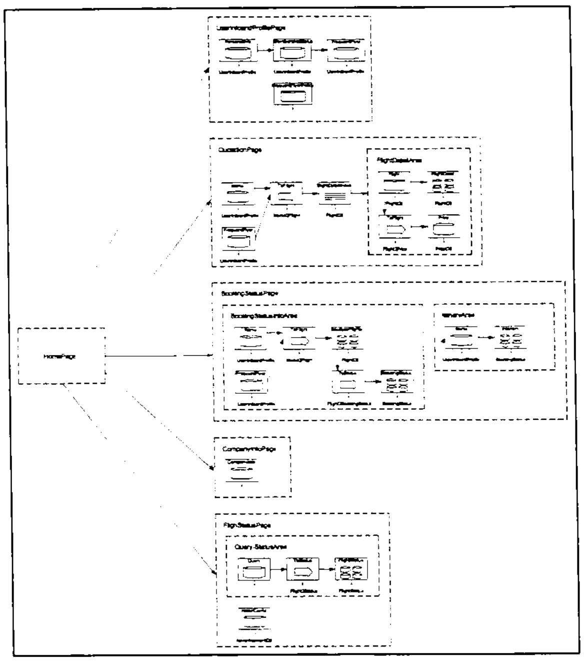

**Figure 5. Example (partial) WebML site view derived from a WebML+ model**

3. Refine the WebML model following the WebML design concept (see [4] for more information on design concept).

As described above we can reverse the $e^3$-value to WebML+ process to create a series of steps for identifying the impact on the business model of changes to the architecture. In a similar way we can also create a series of steps that identify changes to the WebML+ architectural model that arise from changes in the WebML detailed design (though space limitations preclude us from including the detailed steps). The key issue that arises in this process is that WebML models do not explicitly specify actors (or the interactions with these actors), and so either identifying actors, or the interaction with them, in the WebML+ model can still be problematic and would typically require care.

## 6 Conclusion and future work

In this paper, we have demonstrated how WebML+ (a high-level specification language for defining Web system information flows) can be used to form a bridge between business modelling (represented using the $e^3$-value business modelling notation) and low-level information modelling (represented using the WebML notation). We argue that WebML+ provides a clearer connection between the different modelling levels and will subsequently lead to an improvement in Web system development, particularly in terms of the creation of detailed system designs that more accurately reflect business needs. We also believe that this will assist developers and clients in understanding the impact on business processes and models of changes in system designs. In this work, we have also proposed guidelines to support the derivation of a suitable WebML model from/to the WebML+ model and the mapping the $e^3$-value business model from/to the WebML+ model.

Ongoing work is focusing on refining the model (including understanding different types of information provision and derivation) and improving the relationships with the business models, business processes and detailed design. We have also undertaken experiments of the model aimed at evaluating the extent to which it assists developers in understanding the impacts on business models and processes of changes that are made to the underlying designs. We expect that the use of WebML+ will facilitate more rapid and more complete modifications as compared to the use of no modeling notation and the use of existing Web Modeling notations. The outcomes of this evaluation will be reported elsewhere.

## References

[1] L. Baresi, F. Garzotto, and P. Paolini. Extending UML for modeling web applications. In *34th Hawaii International Conference on System Sciences*, Hawaii, USA, 2001.

[2] H. Baumeister, N. Koch, and L. Mandel. Towards a UML extension for hypermedia design. In *«UML» 1999: The Second International Conference on The Unified Modeling Language*, pages 614–629, Fort Collins, Colorado, USA, 1999. IEEE.

[3] G. Booch, J. Rumbaugh, and I. Jacobson. *The Unified Modelling Language User Guide*. Addison-Wesley, 1999.

[4] S. Ceri, P. Fraternali, and A. Bongio. Web modeling language (WebML): a modeling language for designing web sites. In *Proceedings of WWW9 Conference*, Amsterdam, 2000.

[5] J. Conallen. *Building Web Applications with UML*. Addison Wesley Object Technology Series. Addison-Wesley, 2nd edition, 2003.

[6] S. de Cesare, M. Lycett, and P. D. Business modelling with uml: Distilling directions for future research. In M. G. Piattini, J. Filipe, and J. Braz, editors, *4th International Conference on Enterprise Information Systems*, volume IV, pages 570–579, Ciudad Real, Spain, 2002. Kluwer Academic Publishers.

[7] O. De Troyer and C. Leune. Wsdm: A user-centered design method for web sites. In *7th International World Wide Web Conference*, pages 85–94, Brisbane, Aust, 1998. Elsevier.

[8] A. Dutoit and B. Paech. Supporting evolution: Rationale in use case driven software development. In *International Workshop on Requirements Engineering: Foundations of Software Quality (REFSQ'2000)*, 2000.

[9] J. Gordijn and J. Akkermans. $e^3$-value: Design and evaluation of e-business models. *IEEE Intelligent Systems*, 16(4):11–17, 2001.

[10] J. Gordijn, J. Akkermans, and J. v. Vliet. Business modelling is not process modelling. In *ECOMO 2000: published as Conceptual Modeling for E-Business and the Web, LNCS 1921*, pages 40–51, Salt Lake City, USA, 2000. Springer-Verlag.

[11] M. Haverty. Information architecture without internal theory: An inductive design process. *Journal of the American Society for Information Science & Technology*, 53(10):839, 2002.

[12] R. Hennicker and N. Koch. Systematic design of web applications with UML. In K. Siau and T. Halpin, editors, *Unified Modeling Language: Systems Analysis, Design and Development Issues*. IDEA Group Publishing, 2001.

[13] T. Isakowitz, E. Stohr, and P. Balasubramanian. RMM: A methodology for structured hypermedia design. *Communications of the ACM*, 38(8):34–44, 1995.

[14] D. Lange. An object-oriented design method for hypermedia information systems. In *HICSS-27: Proc of the Twenty Seventh Hawaii International Conference on System Sciences*, Maui, Hawaii, 1994.

[15] OMG. OMG unified modeling language specification, version 1.4, September 2001.

[16] L. Rosenfeld and P. Morville. *Information Architecture for the World Wide Web*. O'Reilly, 2nd edition, 2002.

[17] D. Schwabe and G. Rossi. The object-oriented hypermedia design model. *Communications of the ACM*, 38(8):45–46, 1995.

[18] T. J. Shelford and G. A. Remillard. *Real Web Project Management: Case Studies and Best Practices from the Trenches*. Addison Wesley Professional, 1st edition, 2002.

[19] R. Tongrungrojana and D. Lowe. WebML+: A web modeling language for modelling architectural-level information flows. *International Journal of Web Engineering and Technologies*, 2003 (submitted).