A comprehensive design model for integrating business processes in web applications

Damiano Distante*

Research Centre on Software Technology (RCOST)
University of Sannio
Viale Traiano, Palazzo ex Poste
82100 Benevento, Italy
E-mail: distante@unisannio.it
*Corresponding author

Gustavo Rossi

LIFIA, Facultad de Informática
University of La Plata
Calle 50 y 115. (1900) La Plata
Buenos Aires, Argentina
E-mail: gustavo@sol.info.unlp.edu.ar

Gerardo Canfora

Research Centre on Software Technology (RCOST)
University of Sannio
Viale Traiano, Palazzo ex Poste
82100 Benevento, Italy
E-mail: canfora@unisannio.it

Scott Tilley

Department of Computer Sciences
Florida Institute of Technology
150 W. University Blvd.
Melbourne, FL 32901, USA
E-mail: stilley@cs.fit.edu

Abstract: Web applications have evolved from simple read-only websites to complex data- and operation-intensive systems. The main goal of this kind of application is to provide the users with services that assist them in carrying out activities according to a given set of business rules. The addition of transactions to modern web applications poses new challenges, such as managing the interplay between business process execution and navigation, and improving the user's experience in accessing the services that the web application offers. This paper presents a comprehensive design model for integrating business processes in web applications. The model is based on UWAT+, an extended and revised version of the Ubiquitous Web Applications (UWA) Transaction Design model for designing web transactions. UWAT+ makes it possible to

design web application transactions according to the user's perspective and to integrate the web transaction design with the information and navigation design of the web application.

Keywords: web applications; business processes; conceptual design; Ubiquitous Web Applications (UWA); UWAT+.

Reference to this paper should be made as follows: Distante, D., Rossi, G., Canfora, G. and Tilley, S. (2007) 'A comprehensive design model for integrating business processes in web applications', *Int. J. Web Engineering and Technology*, Vol. 3, No. 1, pp.43–72.

Biographical notes: Damiano Distante is a Post-Doctoral Research Associate at the Research Centre on Software Technology (RCOST) at the University of Sannio in Benevento, Italy. He received his PhD from the University of Lecce, Italy in 2004. His primary research area is software engineering, focusing on the reengineering and evolution of software systems, the design and development of large-scale web applications, and the engineering of service-oriented systems. He was the General Chair for WSE 2006, Publicity Chair for WCRE 2006, and has served as Program and Organizing Committee member for conferences and workshops such as *ACM SIGDOC* and *IEEE STEP*.

Gustavo Rossi is a Full Professor at the Facultad de Informatica in the University of La Plata, Argentina where he heads LIFIA, a computer science research laboratory. He received his PhD from PUC-Rio, Brazil, where he developed the OOHDM, one of the mature web design methods. His current research interests include advanced separation of concerns for web applications, physical hypermedia, and volatile requirements in web software.

Gerardo Canfora is a Full Professor of Computer Science and the Director of the Research Centre on Software Technology (RCOST) at the University of Sannio in Benevento, Italy. He was Program Co-Chair for IWPC'97, ICSM 2001, CSMR 2003, and IWPSE 2005, General Chair for CSMR 2003 and WCRE 2006, and is Program Chair for ICSM 2007. His research interests include software maintenance and reverse engineering, service-oriented software engineering, and experimental software engineering. He was an Associate Editor of *IEEE Transactions on Software Engineering* and he currently serves on the Editorial Board of the *Journal of Software Maintenance and Evolution*.

Scott Tilley is an Associate Professor of Software Engineering in the Department of Computer Sciences at the Florida Institute of Technology. He also holds a cross-appointment in the College of Business as an Associate Professor of Management Information Systems. He is a Visiting Scientist at Carnegie Mellon University's Software Engineering Institute. He has a PhD from the University of Victoria. He is Chair of the Steering Committee for the *IEEE Web Site Evolution* (*WSE*) series of events, immediate Past Chair of *ACM SIGDOC*, and General Co-Chair for the *IEEE ICSM 2008 Conference*, which takes place in Beijing, China.

1 Introduction

Web applications have rapidly evolved over the past few years, from brochure-like read-only websites, mainly intended to deliver information and static contents to the user, to complex data- and operation-intensive applications. From 'relatively simple' e-commerce websites to complex web-based information systems, a wide range of web applications rely on the execution of transactions in accordance with certain business processes.

In this context, a *transaction* (or *web transaction*) is defined as a sequence of *activities* that the user executes through the web application in order to carry out a task or fulfil a goal. The set of activities, their properties and the rules governing their execution depend on the business process the application is supposed to realise. The checkout process in an e-commerce site is a simple example of a transaction or process. During checkout, the customer undergoes a set of activities, such as providing his personal data, data about his credit card, address and shipping method. Check-out must be performed step by step and some of the activities might spam others (*e.g.*, if the user changed his preferred credit card, a validation should be done).

Because of the inherent complexity of transactions (as defined above), their addition to modern web applications presents the designer with new challenges and issues to be resolved. A web transaction has a state that needs to be managed on the basis of user content navigation and the execution of activities. Transactions are typically accomplished over several sessions and at times by means of different access devices. The order in which the user is required to execute the activities involved in the transaction (the execution flow of the transaction) and the properties assigned to each of these can impact considerably on the user's experience when using the application. These are just a few of the issues the developer has to deal with when introducing transactions into web applications. These issues require transactions to be properly designed before being implemented. Treating transactions as a by-product of the web application development process usually results in poor usability and erroneous behaviour (Baresi *et al.*, 2002). Implementing a business process just as a chain of navigation steps might introduce further problems as shown in Schmid and Rossi (2004). Web transactions therefore need to be given top priority during the entire web application design process.

A number of models and methodologies formerly proposed for the design of simple hypermedia and web applications have recently been extended in order to provide the designer with a way of tackling the design of transactions. Some of these are briefly described in Section 4 of the paper. In particular, the Ubiquitous Web Applications (UWA) design framework (UWA Consortium, 2001a; 2002) explicitly includes a design activity (named Transaction Design (UWA Consortium, 2001d)) specifically focused on the conceptual design of transactions that implement a business process in a web application. The Transaction Design activity relies on a high-level transaction modelling language, the Unified Transaction Modeling Language (UTML) (Gioldasis and Christodoulakis, 2002) and is able to describe a transaction from both a static (hierarchical organisation) and dynamic (chronology and business logic rules) point of view.

The UWA Transaction Design Model has a solid theoretical basis (drawing from the theory of database transactions (Chrysanthis and Ramamritham, 1994; Elmasri and Navathe, 1998)); it adopts a UML-based notation and boasts other characteristics that make it one of the most useful and effective available models in the literature. However, in a number of reverse modelling and design case studies (Fischetto, 2004), some major weaknesses in the design model have been observed that provide opportunities for enhancement. One of these weaknesses is that the UWA approach is not properly user-centred (contrasting with the overall nature of the UWA design framework). Another drawback is its poor integration with the information and navigation design of web applications designed with UWA.

This paper presents a comprehensive design model for integrating business processes in web applications. The model is based on UWAT+ (Distante, 2004), a revised and extended version of the UWA Transaction Design Model for designing web application transactions, which overcomes the weaknesses experienced in the original model. The UWAT+ approach is the result of a thorough analysis of the set of requirements a methodology supporting the design of business processes in web applications must fulfil, and the evaluation of different methodologies (in particular UWA) against these requirements.

The main contributions of this paper are the following:

- The presentation of the UWAT+ conceptual model together with an example of use for a representative web application implementing a business process.
- As a by-product, the enunciation of a set of requirements for methodologies supporting business processes in web applications.
- The assessment of the most relevant web design methods against the list of requirements.

The rest of the paper is organised as follows. Section 2 presents an example that motivates the research. Section 3 enumerates and explains the set of requirements for a method supporting the design of business processes in web applications. Section 4 briefly discusses a number of works related to ours. In particular, the approaches to designing web transactions adopted by some of the best-known hypermedia application design methodologies are described and assessed with respect to the requirements. Section 5 presents UWAT+ describing its conceptual reference model and highlighting the major changes and extensions with respect to the UWA Transaction Design model on which it is based. Section 6 describes the adoption of UWAT+ in the process of designing a web system for managing the process of competitive tendering for an Italian Local Public Health Company (AUSL). Section 7 discusses the main features of the UWAT+ method. Section 8 presents the concluding remarks and possible avenues for future work.

2 A motivating example: the process of competitive tendering of an Italian local public health company

The competitive tendering process in all public administration (including public hospitals) is part of a larger process of procurement that starts by analysing and defining the services and goods representing the procurement needs, includes the tendering

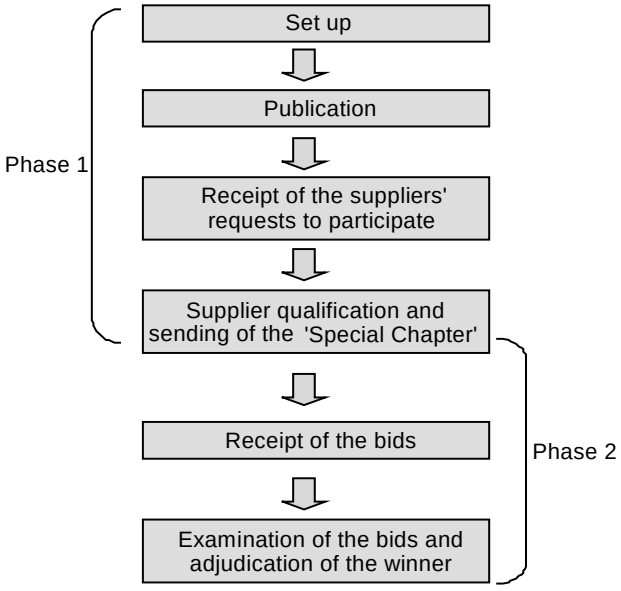
process, continues with the ordering/providing process and ends with the warehousing and usage of the bought goods and services by the organisation. E-procurement has been stimulated by most of European governments (including Italy) for several years.

The process of competitive tendering of an Italian local health public company (AUSL) is used as an example to illustrate the issues that a designer has to cope with when designing a business web application (*i.e.*, a web application implementing business processes) and the requirements that can be derived for a candidate design methodology. The remainder of this section briefly describes the process of competitive tendering and introduces the design issues it carries out. A set of requirements for a suitable design methodology of web applications implementing business processes is described in Section 3.

2.1 An overview of the competitive tendering process

Figure 1 depicts the process of competitive tendering for an Italian local health public company (AUSL) from the perspective of the competitive tendering manager. The process can be conceptually divided into two phases: the phase of setting-up the call for tenders (Phase 1), and the phase of participating and adjudicating of the winner of the competitive tendering (Phase 2).

Figure 1 The competitive tendering process of an AUSL (an Italian local health public company) from the competitive tendering manager actor point of view



The six main activities that make up the process are the following:

1 Setting up the call for tenders

A call for tenders is set up by an AUSL when there is a need to organise one or more contracts for the procurement of pharmaceutical and para-pharmaceutical products. The competitive tendering manager establishes the details of the tender, specifying

the conditions of the contract, as well as the guarantees that the competing companies must provide (technical and legal requirements) and the various procurement needs that will be included in the tender.

2 Publication of the call for tenders

In this phase the announcement with the relative calls to tender is published in local and national daily newspapers, as well as in the *Official Journal of the European Union* in order to have a high number of potential providers participating.

3 Receipt of the requests to participate in the competitive tendering

When the suppliers have read the announcement, they may show their intention to compete for the contracts by sending a formal request to be admitted to the competitive tendering.

4 Examination of the requests received and sending of the Special Chapter to the admitted suppliers

The health company examines the requests received, and draws up a list of the suppliers admitted to the tender. These companies alone will then be sent the 'Special Chapter' containing the list of the Items (or of the lots) being put out to tender, the quantity required for each item, the evaluation criteria for the articles offered by the bidders, the procedure for the compilation of the bids and the criteria of adjudication.

5 Formulation of the bids by the suppliers

The suppliers admitted to the tender read the Special Chapter and may formulate their bids, specifying the description and the quantitative, qualitative and economic details of the product on offer.

6 Examination of the bids received and adjudication of the winner

When the closing date of a tender arrives, the next step is the evaluation of the bids received. Depending on the type of tender, the evaluation may be simply economic or may be accompanied by a technical evaluation (of quality with respect to the technical parameters of evaluation that were set down) of the articles on offer. After the evaluation of the bids received, the winning bidders are selected and an official graded list is published.

2.2 Design issues

Some of the major issues the designer has to cope with when designing the web application at a conceptual level to support the above described business process are presented in the following:

DI1 The execution of the process of competitive tendering requires a number of activities to be carried out respecting a set of business rules and temporal constraints. In designing the web application that has to support the business process, the designer has to take into account the activities in which the business process is structured and their relations, both hierarchical and logical.

DI2 A number of different actors are involved and collaborate in the execution of the competitive tendering process. Actors internal to AUSL include the *General Director* of the AUSL, the *Heads of the AUSL Hospital Units*, the *Competitive Tendering Manager* and the *Technical Commission for Evaluation*. Actors external of AUSL include the *Suppliers* that aim to participate at the competition tendering with an offer (bid) and the *Press Agents* that will advertise the call for tender. Sometimes a complex activity requires the interaction of more actors; more often for an actor to play his role in the process, previous activities have to be completed by other actors, according to the business rules that apply to the process model. The design of the web application has to consider the set of actors involved in the process (most of which become types of user of the application), the needs of each of them for executing the activities they are in charge of, and the way they need to interact in executing the process.

DI3 In the competitive tendering process, a high volume of data and information are involved and need to be managed. As in any other web application, there is a need for designing the structure of the information according to the way the user will access and navigate this information. Differently from purely informative web applications, data and information in a web application implementing business processes have to be related to the activities of the process. Navigation and operation execution are interleaved and the effect of the one on the other has to be defined. As an example, when setting up the call for tender, the manager of the competitive tendering has to specify the goods and the services needed by the AUSL, the characteristics of each of them, the way and deadlines for providers to submit their bids, the contractual conditions the winners of the competitive tender have to subscribe to and obey when providing the goods or services they have won, and so on. The competitive tendering manager needs to access and surf all these data and use them to set up the call for tender. Different data and information are needed for any different activity of the process.

DI4 The process of competitive tendering may last several months, from the moment it is started and the call for tender published and open, to the moment the winning providers are defined and the AUSL can start issuing orders for provision. Also one of the activities included in the process may need a long time, and basically several sessions of usage of the supporting web application, to be carried out. This implies the need for the state of the ongoing transaction and of an activity that the user carries out in several sessions to be stored and used at each following session.

DI5 The navigational nature of the web has to be coupled and managed with the operational and transactional nature of business management applications as e-procurement systems. As an example, when examining the offer of a supplier participating at the competitive tendering, the members of the technical commission that have to evaluate and assign a score to the offer need at the same time to read the description of the offer and compare it with the request and the characteristics defined in the call for tender. Then, express a score for each of the evaluation items defined in the call for the considered offer. Carrying out this activity, from the web application point of view, requires the content navigation and operation execution inside each activity of the process to be designed in a comprehensive way. Should the user be enabled or suggested to navigate towards

certain contents while executing a certain activity? And if the answer is yes, what should happen to the state of the ongoing process or activity? On the other hand, when executing an activity, which contents should be provided and which navigation step should be followed? Which contents should be associated and provided to the user to support him in executing an activity?

DI6 The experience that the users have when using a software system is one of the crucial factors determining the success of the system. A business management web application is mainly intended to enable the user to execute operations and carry out activities of a business process. The capability of the application to effectively support the user in carrying out these activities is an essential property determining the quality of this kind of web applications, as perceived by its users. Designing the application from a user point of view, and taking care of the users' expectation are issues the designer has to cope with.

DI7 Ubiquity (*i.e.*, the possibility to access the application by using different types of devices and being in different context of usage) could be an important property to provide to the system to be designed. In the case of the competitive tendering example, a requirement could be that the competitive tendering manager should be enabled to access the system and check the status of the tender and have synthesis data of the process.

Additional concerns involve for example privacy to protect data on offers and suppliers, security to avoid hacking, *etc.* However, these concerns usually belong to a lower level of abstraction in the development process and therefore are not treated in this paper. As a concluding remark on the design issues presented above, it should be easy to accept that they are not specific of the example process considered but apply to any other business process.

3 Requirements for designing business processes in web applications

Moving from the experiences gathered in a number of case studies of reverse engineering and forward design of web applications implementing business processes (Fischetto, 2004), and from the analysis and comparison of the approaches adopted by a number of design methodologies such as OOHDM (Schmid and Rossi, 2004; Schwabe *et al.*, 1999), OO-H, UWE (Koch *et al.*, 2003), WSDM (De Troyer and Leune, 1998; De Troyer and Casteleyn, 2003), ADM (Atzeni *et al.*, 1997; Atzeni and Parente, 2001) and UWA (UWA Consortium, 2002; 2001a), a representative set of requirements for a suitable methodology and related models for designing business processes in web applications was built. In the following, each of the requirements is formalised and contextualised with the list of design issues identified for the competitive tendering business process defined in Section 2.1.

Req1 Identify the component activities of a web transaction, their semantic associations, and the properties/constraints that apply to each of them.

Web transactions are a way to implement the computerisable portions of a business process in a web application. Therefore, the design of web transactions has to start considering the set of activities included in a business process and

has to observe the semantic and temporal relationships existing between them. A suitable design methodology should provide the designer with a way to represent the organisation of a web transaction in terms of its component activities, their semantic relations and the properties and constraints that apply to the execution of each of them. As an example, an activity can be defined alternative to another, or can support the user to execute another activity. Also an activity can be required or optional to complete the execution of the web transaction. Complex activities may be split in smaller and simpler activities, and may be suspended for long-lived transactions.

Req2 Describe the possible execution flows (or workflows) of a web transaction.

A business process has associated with it one or more workflows describing the order in which its component activities can be executed and the conditions under which each activity can be executed. A suitable design methodology for designing business processes in web applications should enable the designer to represent these workflows, describing which activities need to be executed in a particular order and which can be executed simultaneously, with no precedence rule as regulated by the underlying business process. Also, the conditions to be true for the user to be enabled to execute a given activity (the user must be logged, *etc.*) and the changes caused by the execution of the activity to these conditions have to be described. These conditions have to be stored and maintained to represent the state of a transaction with regard to a specific user.

Req3 Define and manage the state of a web transaction.

As stated in requirement Req2, to check the conditions under which a given activity of an ongoing transaction can be executed, the related information has to be stored and maintained during all the execution time of the transaction, for a given user. A suitable methodology for designing business processes in web applications should clearly define the concept of 'state' of a web transaction and give the designer a way to describe how it changes over the execution of the web transaction by a given user. Once it has been defined, the state can be used to describe the rules that apply to the execution of a given component activity inside the workflow of the transaction as well as to store the conditions in which a transactions is suspended when the user leaves it for resuming afterwards. Such information may include the identifier of the user executing the transaction, the set of activities already executed and the activity to start from when the transaction resumes.

Req4 Specify which activities can be suspended and resumed afterwards during long-lived transactions.

This requirement is related to the need for the designer to represent long-lived transactions, *i.e.*, transactions that are completed through more than one session and (in ubiquitous applications) in a number of different contexts of executions (different access devices, locations, time, *etc.*). Within the constraints entailed by the business rules of the implemented business process, the design

methodology should enable the designer to describe if and when a web transaction can be suspended in order to be resumed afterwards, that is, which of the component activities are suspendable and which are not.

Req5 Describe the way two or more types of users involved in a web transaction collaborate in its execution.

Usually, more than one actor are involved in a business process and collaborates on its realisation. As a consequence, in a web transaction implementing a business process, typically more than one type of user of the web application are involved. A suitable design methodology should explicitly provide the designer with a way to describe how the types of user involved in a web transaction collaborate on its execution.

Req6 Specify the way content navigation and operation execution affect each other and the state of an ongoing web transaction.

As discussed in Section 1, both content navigation and operation execution are involved and mixed together in a web transaction. A web application is naturally based on content navigation. On the other hand, a business process requires the user to execute a certain number of operations providing input data and making choices. A suitable model for designing web transactions should enable the designer to specify which navigation is associated with the execution of a given operation and which state transitions of a web transaction are caused by the user following a given navigation link.

Req7 Define which contents will be provided to the user in order to support the execution of a particular activity.

A suitable web transaction design methodology should enable the designer to specify which contents should be provided and, eventually, which information will be requested from the user when executing each of the component activities of a web transaction. The designer should also be able to define which information objects are affected by the execution of the activity.

Req8 Define which information objects are affected by the execution of the activity and how.

A functional activity consists of the execution of one or more elementary operations (insertion, deletion, modification, *etc.*) on the data stored and manipulated by the application and the information objects involved in the activity. A web transaction design model should enable the designer to define which operations are fundamental to each activity, thus modelling the way each elementary activity affects the information objects it involves, modifying their instances because of its execution. Finally, each input operation on the data of an information objects creates a new instance of that object, thus changing the contents of the application.

Req9 Describe the way an activity will be customised depending on the state of the ongoing transaction.

An example to clarify this requirement, as also discussed in Schmid and Rossi (2004), is the following: consider the activity of 'add to cart' in an e-commerce website. What should be the effect of executing this activity two or more times for the same product? Should the quantity of the product included in the cart be increased each time? Should the system inform the user of the repetition of the activity? Should the 'add to cart' button be disabled for a product once included in the user's shopping cart? Another common situation is the one related to forms that have been filled in by the user and that the user is required to fill in again: should the previous data specified by the user be showed to the user or should the form appear empty of data? There is no unequivocal solution to these particular issues, but a suitable web transaction design model that helps the designer to consider it and to make conscious choices is surely helpful.

Req10 Describe the way an activity will be customised depending on the context of execution.

The growing number of types of access devices (PC, PDA, mobile phones, *etc.*), of communication technologies (internet, WAP, UMTS, *etc.*) and the high mobility of the users make web applications that are ubiquitous more desirable every day. As the need for ubiquitous web applications grows, web application design methodologies should enable the designer to design a web transaction with regard to the possible contexts of execution (device, location, time, type of user, user profile, *etc.*).

Though not exhaustive, the above list of requirements is representative of the needs of designers when specifying web transactions in a web application. Their definition resulted from the analysis of different approaches proposed by well-known design methodologies and the experience gathered in developing a number of case studies of reverse and forward design of business web applications.

Table 1 provides the mapping of each of the design issues identified in Section 2.2 for the motivating example, onto the list of requirements presented above.

Table 1 Mapping of the design issues identified in the motivating example of Section 2.2 onto the list of requirements presented in Section 3

Design issue	*Requirements*
DI1	Req1, Req2
DI2	Req3, Req4
DI3	Req6, Req7
DI4	Req3, Req4
DI5	Req5, Req6, Req8
DI6	Req7
DI7	Req10

4 Related work

The relevance of business process design in web applications is shown by the evolution that many web application design methods have undergone in the last few years. Well-known web application design methods such as ADM (Atzeni and Parente, 2001), OO-H (Koch *et al.*, 2003), UWE (Koch *et al.*, 2003), WSMD (De Troyer and Leune, 1998; De Troyer and Casteleyn, 2003), OOHDM (Schmid and Rossi, 2004) and UWA (UWA Consortium, 2002) now include approaches, concepts and models specifically addressing the design of business processes. In the following, the approach adopted by each of the mentioned methods is described.

4.1 UWA

The Ubiquitous Web Applications (UWA) design framework provides a complete methodology and a set of models and tools for designing ubiquitous web applications. The UWA methodology includes a design activity named Transaction Design, specifically intended for designing web transactions, *i.e.*, business processes in web applications. The model resulting from this design activity makes use of three main concepts: Operation, Activity and Execution Contract. An Operation is defined as a non-suspendable atomic unit of work that cannot be further decomposed. An Activity is defined as a set of operations and possibly other activities with an optional flow of execution, *i.e.*, an optional order of execution to be respected between them. The Execution Contracts are the properties of an activity that characterise its type. For each of the previous primitives, a notation and a set of properties are provided. A web transaction corresponds to a complex activity composed of sub-activities with an associated execution flow and is basically designed by means of two models, the Organization Model and the Execution Model.

The Organization Model, besides modelling the hierarchical relations between activities and sub-activities, also specifies whether the execution of the sub-activity is required or optional in order for the user to complete the ancestor activity, or if changes to data resulting from the execution of a sub-activity are visible by other concurrent users.

The Execution Model is intended to define the possible execution flows among the activities that compose the transaction and are included in the Organization Model. In this model, activities and sub-activities are represented by states (ovals), and the execution flow between them is represented by state transitions (arcs). The model shows the state of a web transaction from the system point of view and the execution rules (chronological and logical) that apply to each activity.

4.2 OOHDM

The Object-Oriented Hypermedia Design Model (OOHDM) is an object-oriented method for web application design (Rossi *et al.*, 2003). In order to model the User Activity, also referred to as the Business Process, equivalent to the web transaction in UWA, the OOHDM conceptual model was modified by adding two concepts: processes and activities. Processes can be considered part of the conceptual model and are related to the

execution flow of the entire transaction, while Activity Nodes can be thought of as part of the Navigation Model and are the means by which the interaction between navigation and transaction execution is designed.

From a static point of view, an activity can be broken down into two or more sub-activities (also referred to as Basic Activities) to form a hierarchical structure, represented with UML class diagrams. From a dynamic point of view, an activity has a flow control modelled with the UML activity diagrams. Some of the most interesting aspects of the OOHDM Business Process Design model are suspendable activities, navigation inside a process context, activities that are executable in parallel, and customisable processes.

4.3 WSDM

The Web Site Design Method (WSDM) is a user-centred method for web application design. The design process consists of four different phases: Mission Statement Definition, Audience Modelling, Conceptual Design (divided into Task Modelling and Navigation Design) and Implementation design.

During the Task Modelling phase, all the information and functional requirements obtained during the Audience Modelling phase are elaborated to generate two main models: the Task Model and Object Chunks. In particular, for each requirement a task is defined, and then divided into elementary tasks, in a hierarchical structure. This structure is described by means of a Concur Task Tree (CTT) (Patern *et al.*, 1997), a special notation used in the field of Human-Computer Interaction (De Troyer and Casteleyn, 2003). Every process (task) identified is a leaf of the tree and is related to one or more processes on the same level by means of a structural relationship.

The main purpose of the Navigation Design phase is to define a conceptual structure for the web application and to model the navigation, which each class of users is allowed to carry out. The result of this design step is the Task Navigation Model that describes how a user can perform a task during the execution of the application.

4.4 UWE

The UML-based Web Engineering (UWE) approach is a UML-compliant method of web application design, which is divided into six phases: Requirement Analysis, Conceptual Model, Process Model, Navigation Model, Presentation Model and Architectural Model. Because the Conceptual model is not able to provide information about the underlying business processes that drive users through the application, a specific design activity was introduced to cope with this problem: the Process Design, also called Task Modelling. During this design step, a web process is broken down into several sub-activities in a hierarchical way.

The user activities are modelled by means of UML activity diagrams, which can be enriched with input/output objects from crucial activities. At the end of this phase, the Navigation Model is enriched by a set of integration models, which represent all the points in the application from which the user leaves navigating through the content and starts the execution of a business process.

4.5 WebML

WebML is a model-based approach for data-intensive web applications, with an associated tool support called WebRatio (Ceri *et al.*, 2000). It partitions the design space basically in three models: a data model (corresponding to conceptual models of previously described methods), a hypertext model (analogous to formerly described navigation models), and a presentation model. WebML has been extended to support process modelling by extending the data model with process meta-data and enriching the hypertext model with process enactment primitives (Brambilla, 2003; Brambilla *et al.*, 2006). Data model meta-data (expressed using the popular entity/relationship model) support the Workflow Management Coalition process model (WfMC, 2006). The new hypertext semantics facilitates the indication of the start and end of activities, the expression of sequences, and the description of process-aware content units. The notation of the WebML process primitives is expressive and rich.

4.6 Discussion

Each of the above-listed design methods has been evaluated and compared to the list of requirements defined in Section 3 with regard to the approach they adopt for designing business processes. Results of the evaluation are summarised in Table 2. The study showed that none of the above-considered design methods completely fulfil each of the requirements of the set. In particular, OOHDM has been found to be the method that best performs the set of requirements.

Table 2 Design methods evaluated over the list of requirements presented in Section 3

		Methods				
		UWA	*OOHDM*	*WSDM*	*UWE*	*WEBML*
Requirements	RQ1	◐	◐	●	○	●
	RQ2	◐	●	●	●	●
	RQ3	●	●	○	●	◐
	RQ4	●	●	○	●	◐
	RQ5	○	○	●	◐	◐
	RQ6	○	●	○	●	◐
	RQ7	○	●	●	○	●
	RQ8	○	○	●	○	○
	RQ9	◐	●	○	○	○
	RQ10	●	●	◐	○	○

Notes: ○ Requirement not satisfied
◐ Requirement partially satisfied
● Requirement fully satisfied

5 UWAT+: an extended version of the UWA transaction design model

Although UWA supports the design of the content and navigation structure of the application, major shortcomings have been experienced in using its transaction design approach in a number of case studies (Distante *et al.*, 2004; Distante and Tilley, 2005). As shown in Table 2, UWA does not satisfy requirements RQ5, RQ6, RQ7 and RQ8. Indeed it appears that the UWA Transaction Design method and its related models:

- Do not offer a way to describe the interplay between the execution of process activities and the navigation of application contents.
- Performs poorly in specifying which content and which information objects are involved or necessary when the user executes an activity.
- Even if UWA is a multi-user design methodology, it does not explicitly propose a way to design how different user types may collaborate in a business process.

It is somewhat evident that the needs related to requirements RQ6-RQ8 might be addressed by the UWA Operation Design (UWA Consortium, 2001c). However, the granularity in which this would be done during this design activity would be too small and also the adopted specification (pre- and post-conditions specified in OCL) is too heavy to be adopted for each operation the user may invoke in a given activity. On the other hand, designing web transactions with UWAT+ obviates (unless a very fine-grained level of design is desired) the execution of the UWA Operation Design.

The knowledge gathered analysing and comparing design methods alternative to UWA, such as the ones examined in Section 4, led us to propose an extension of the UWA transaction design that overcomes the noticed shortcomings. This extended version of the UWA transaction design model is called UWAT+ (Distante, 2004). Besides addressing the noticed shortcomings, UWAT+, compared with its original version, also boasts a user-centred design approach and is better integrated with the remainder of the UWA design process.

The rest of this section is dedicated to the presentation of UWAT+, describing its conceptual model and highlighting what is new to the original version defined in UWA. The application of UWAT+ to design one of the transactions of the 'competitive tendering' business process described in Section 2 is illustrated in the following section.

5.1 UWAT+ conceptual model

Figure 2 shows the UML class diagram representing the concepts (design primitives) and the relations among concepts defined by UWAT+ (UWAT+ conceptual model, left portion of the diagram) and the models used by UWAT+ to design a web transaction by making use of the concepts (UWAT+ design models, right portion of the diagram).

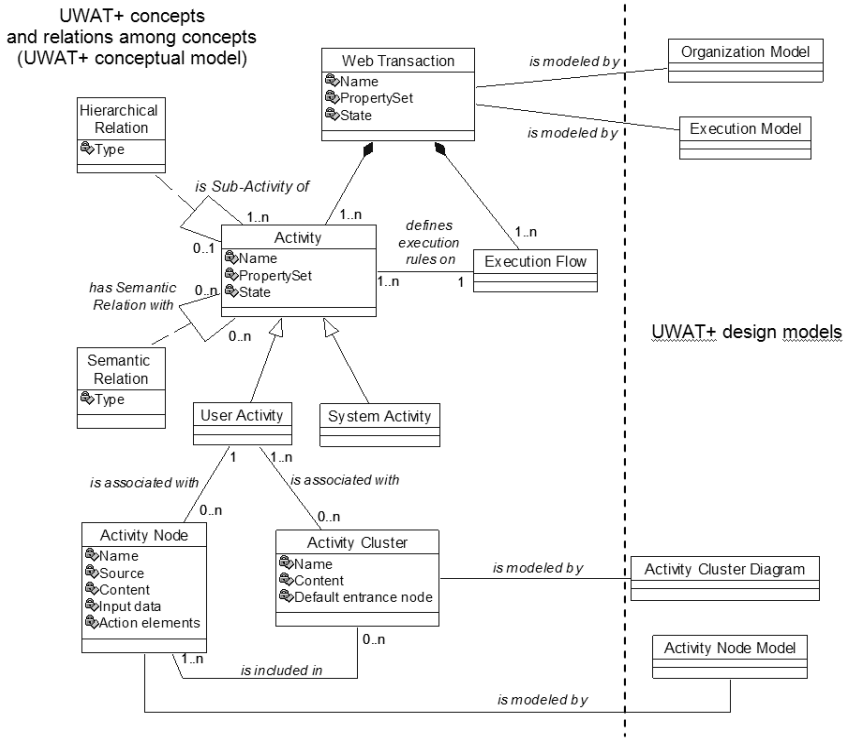
As described by the diagram, a UWAT+ web transaction is a complex object (concept) composed of two main types of objects:

1 Activity – that distinguishes between User Activity, which is a task a user should execute to achieve his/her goal, and System Activity, which is an activity executed by the system without any user interaction.

2 Execution flow – that represents the logical and temporal order with which activities must or should be executed to complete the transaction.

An activity has a Name, a PropertySet, and a State. The PropertySet consists of five properties that an activity may or may not satisfy. These properties are: Atomicity, Consistency, Isolation, Durability, and Suspendability (ACIDS). An Activity is defined as suspendable if it can be stopped during one work session and continued in a subsequent one from the exact point where it was left unfinished. All the information about the transaction execution state is represented by the state variables. More details on this regard can be found in the following sub-sections.

Figure 2 UWAT+ concepts and design models



As illustrated by the conceptual model, a web transaction can be completely described by two models: the Organization Model and the Execution Model. The former represents a web transaction from a static point of view. It models the hierarchical organisation of the activities, which compose the transaction using the Hierarchical relations and Semantic relations existing among them. The latter represents a web transaction from a dynamic point of view, defining the execution rules of the component activities.

Each User Activity can be associated with one or more Activity Nodes and one or more Activity Clusters. An Activity Node allows the designer to specify which contents (drawn from the Information and Navigation Model) must be provided to the user in order to support the execution of an Activity. An Activity Cluster, similar to the UWA Navigation Cluster, specifies the possible navigation between a set of Nodes. In addition,

by specifying the transitions occurring in the state of a transaction when the user navigates towards a Node and the Node the user reaches after executing an Activity, an Activity Cluster defines the interaction between transaction execution and content navigation. Together, the concepts of Activity Node and Activity Cluster allow the designer to uniformly merge the conceptual design of the transaction (Organisation and Execution Models) with the design of the Information and Navigation Models.

A web transaction has a state that is stored in dedicated variables whose value changes depending on transaction execution and content navigation. In our model the state is defined on two levels: a transaction level and an activity level.

The variables that characterise a transaction state are:

- Transaction history – an ordered list of all the activities executed by the user.
- Current activities set – the set of activities that are being executed.
- Execution state – either ongoing, suspended, completed or aborted.

The variables that characterise an activity state in the context of a specific transaction execution are:

- Execution state – this defines the activity state of the last execution; its possible values are either ongoing, suspended, completed or aborted.
- Input variables cache – the values specified by the user for the input variables during the last activity execution. Those values can be used as defaults for the next execution.
- Default input values – optional default values for activity variables.

The next sub-sections describe the models used by UWAT+ to design a web transaction, namely, Organization Model, Execution Model, Activity Node and Activity Cluster diagrams. The most important differences with respect to the original UWA Transaction Design model are also highlighted.

5.2 Organization model

The Organization Model is a customisation of the UML class diagram (Object Management Group, 2004) in which class stereotypes are used to represent activities and sub-activities of a web transaction and their properties, and associations between classes are used to represent relations between activities. An Organization Model represents a web transaction from a static point of view; activities and sub-activities of a web transaction are arranged to form a tree. The main activity represented by the root of the tree corresponds to the entire web transaction, while activities and sub-activities are intermediate nodes and their leaves. Significant changes have been made to the Organization Model by dividing all possible relations between activities into two categories: *Hierarchical Relations* and *Semantic Relations*. Hierarchical Relations hold between an activity and its sub-activities and specify if one or more of the execution of one or more of the sub-activities A_{ij} of a given activity A_i is required or optional to complete the execution of the activity A_i. Possible types of Hierarchical Relations are: *Requires*, *Requires One*, and *Optional*. Semantic Relations may hold between any two

activities of an Organization Model and describe if an activity is of support for the user to execute another activity (*Can Use*), if an activity compensates the execution of another activity (*Compensates*), and so on.

An example of the Organization Model is shown in Figure 5, Section 6.

5.3 Execution model

The Execution Model represents a web transaction from a dynamic point of view. It is a customisation of the UML activity diagram (Object Management Group, 2004) where activities and sub-activities are represented by states (ovals), and the execution flow between them is represented by state transition (arcs).

While the original UWA Execution Model includes both user- and system-design elements mixed together, it is felt that system-related aspects should be specified in a different model. In UWAT+, system activities are still represented as user activities, but it is specified that they are executed by the special user called 'system'.

Each possible transition among activities must be explicitly represented in the model with a line between them, and, where useful, 'tagged' using a simple and extensible labelling mechanism to specify the category and/or some semantics/rule associated with the transition: A (action invoked by user); C (condition(s) required for Activity execution); R (result of the execution of an activity); and S (state associated with the system due to the execution of an activity). A list of the causes of activity failure and possible actions the user or the system can take can also be specified by means of a failure table.

These changes to the Execution Model provide a better view of the dynamic execution paths the user will experience while completing a specific transaction. By making such paths explicit, improvements in the transaction design can be more easily accomplished. UML swimlanes diagrams (Object Management Group, 2004) are adopted to describe how two or more user types of the application collaborate in the execution and completion of a transaction.

An example of the Execution Model is shown in Figure 6, Section 6.

5.4 Activity node diagram

UWA defines a Navigation Node as a unit of information provided to the user in a single step. UWAT+ introduced the concept of Activity Node as an interactive Navigation Node, intended to support the execution of an elementary activity of a transaction. Once the Activity Nodes have been defined, they can be handled (from a navigational point of view) like other navigation nodes and can be used in Navigation Clusters. UWA uses the concept of Navigation Cluster and the associated model to specify the possible navigation paths between a set of nodes in a given context.

An Activity Node makes it possible to specify:

- The relationships existing between the Transaction Model and the Information Model of the application, in terms of which contents are necessary and will be available for the user when executing an activity.
- The actions (corresponding to transitions in the Execution Model) the user can invoke when visiting the node, where useful.

- The effects of the execution of the activity to which the node is associated on the Information objects defined in the Information Model.

UWAT+ Activity Nodes are designed by means of Activity Node diagrams. An example of Activity Node Diagram is shown in Figure 7. An Activity Node is characterised by five attributes: *Name*, *Source*, *Content*, *Input Data* and *Action Elements*. The Source attribute specifies the activity to which the Activity Node is associated. The Content attribute indicates which of the navigation nodes of the Navigation model are included in the Activity Node, thus specifying the contents that will be provided to the user when executing the activity. The Input Data specifies the data that will be requested to the user to execute the activity. The Action Elements define the elements, such as hypermedia links, action buttons, *etc.*, that the user will use to execute the activity or to move to another activity.

5.5 Activity cluster diagram

UWAT+ introduced a new category of UWA Navigation Cluster, the Activity Cluster, which is associated with the activities of the Transaction model. Figure 8 shows an example of Activity Cluster Diagram used to design UWAT+ Activity Clusters. An Activity Cluster has the following characteristics:

- It includes one or more Activity Nodes and specifies the default node the user is presented when entering the cluster.
- It shows the Navigation Links (continuous lines) and Activity Links (dashed lines) among nodes.
- It shows the transaction state modifications associated with each link.

Navigation Links model simple navigational steps, and can connect two Navigation Nodes, a Navigation Node with an Activity Node, and even two Activity Nodes. The latter case means that a user can navigate between the two activities without executing them.

In contrast, Activity Links are associated with the Action Elements of an Activity Node. In accordance with the execution flow described by the Execution Model, they specify the transitions between activities and, at same time, the navigation between nodes when the user invokes the corresponding action. Each Activity Link may be labelled with the Action (corresponding to a User Call in the Execution Model) it is associated with.

Both Navigation Links and Activity Links are labelled to specify how the state of the ongoing transaction and the executing activity are affected when the user follows them. Valid values for the labels are *Ongoing*, *Suspended*, *Completed*, *Aborted* and *Resumed*.

Different Activity Nodes and Activity Clusters may be associated with a certain Activity to design how the activity will be customised, that is, how it will appear to the user, depending on the state of the transaction and the context of the execution of the activity. As such, transaction state takes into account, for example, if the user has already executed the activity during the ongoing transaction.

The context of execution may instead consider variables such as the user profile, the location and time, the access device currently being used, and so on. The different Activity Nodes and Clusters associated with an activity may be modelled as variations of a default Activity Node and Activity Cluster associated with the activity by means

of customisation rules as suggested by the UWA Customisation Design activity (UWA Consortium, 2001e). Customisations may regard the content and the navigation links that are provided to the user when executing the activity, as well as the other attributes included in the Activity Node and Activity Cluster design primitives.

6 Designing business processes in web applications with UWAT+

This section reports the results of a real project in which the UWAT+ model was used to design an e-procurement application for an AUSL (an Italian Local Public Health Company). The steps followed in the design of the application are shown, with a particular emphasis on the 'Management of the competitive tendering' transaction for the user type called 'Competitive tendering manager'. The process of managing a competitive tendering for an AUSL is made up of the six phases shown in Figure 1.

The diagram in Figure 3 shows a customised version of the UWA design process in which UWAT+ is used for the design of the application transactions. The notation used to illustrate the design process is IDEF0 (IDEF-0, 1993). It is important to notice that compared to the original version of the UWA design methodology, the typical order in which the set of design activities are conducted has been slightly changed. In particular, the Navigation design activity follows the Transaction design, and both of these activities make use of the UWAT+ conceptual model. The phases of the design process, as described in (UWA Consortium, 2001a) are:

1 Requirements elicitation

2 Information design

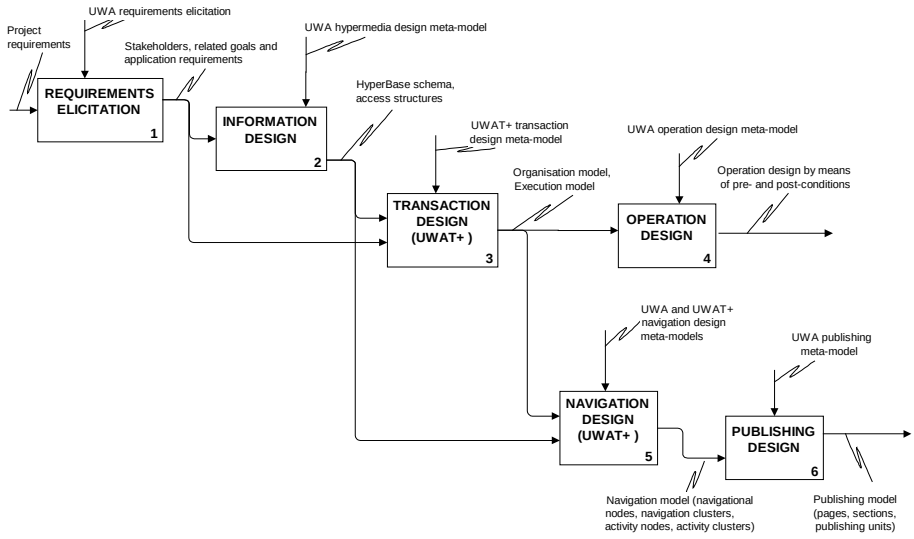
3 Transaction design

4 Operation design

5 Navigation design

6 Publishing design.

Figure 3 The process of design of a web application using UWA and UWAT+



6.1 Requirements elicitation

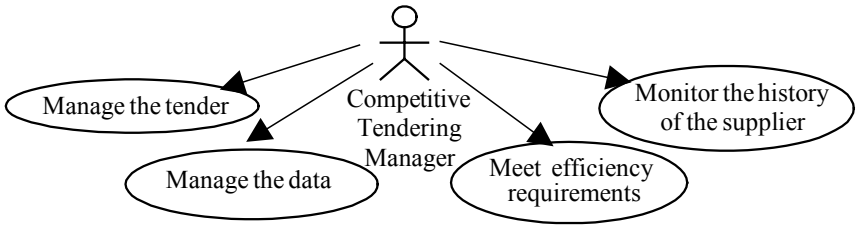
The Requirements Elicitation phase takes as an input the project specifications and produces, by means of the refinement process mechanism, the following output:

- Stakeholders and relative goals
- Application requirements.

The approach used is goal-oriented: each stakeholder owns at least one goal, *i.e.*, an abstract and long-term objective that the system must make it possible to achieve; each goal is then refined into sub-goals, down to the requirements, which are concrete and short-term objectives, on a sufficiently low level to be implemented. This phase is exactly the one described in (UWA Consortium, 2001b). The stakeholders (also users of the application) identified for the case under study are: *Competitive Tendering Manager*, *Technical Manager*, *Supplier* and *Citizen* or *Generic User*.

The competitive tendering manager follows the entire process of management of a competitive tender. Figure 4 shows the high-level goals identified for this user type.

Figure 4 High-level goals for the competitive tendering manager



6.2 Information design

During the Information Design phase (UWA Consortium, 2001c), the contents of the application along with the ways provided to the user to access them are designed. The output of this phase is the hyperbase of the application (a set of Entity types, each describing a class of information objects, and the semantic associations between them) and the set of access structures (view on the hyperbase according to a particular selection criterion).

The next design step is the Transaction Design phase, carried out in accordance with the UWAT+ model presented in Section 5. Starting from the results of the Requirements elicitation, high-level transactional-type goals are selected, *i.e.*, goals that require the execution of one or more activities in order to be fulfilled. For each of these goals a transaction is designed, which will support it firstly from the static point of view by creating the Organization Model and then, from a dynamic point of view, by means of the Execution Model. The Organization Model and the Execution Model both constitute the output of the Transaction design process. The Organization Model of the 'Management of the competitive tendering' transaction for the competitive tendering manager user type describes the transaction from a static point of view in terms of the activities of which it

is composed and the relations between them. The Execution Model specifies the flow of execution of its activities. Using Swimlanes, it shows the way the competitive tendering manager, the technical manager and the supplier will collaborate in this transaction.

Figures 5 and 6 respectively show a portion of the Organisation and the Execution Models which define the conceptual model of the transaction associated with the competitive tendering process. In particular they show the design of the activity called 'Bid analysis and evaluation' with its relative sub-activities. In the Execution Model, full lines and dashed lines respectively indicate mandatory and optional execution flows.

Figure 5 The Organization Model for the activity named 'Bids analysis and evaluation' in the 'Management of the competitive tendering' transaction

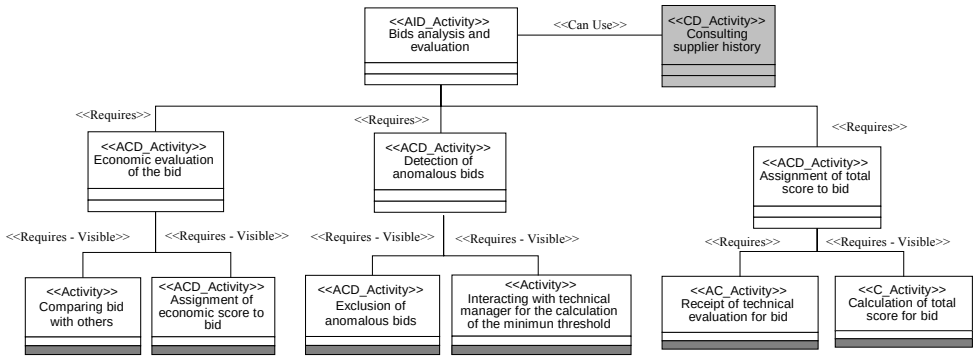
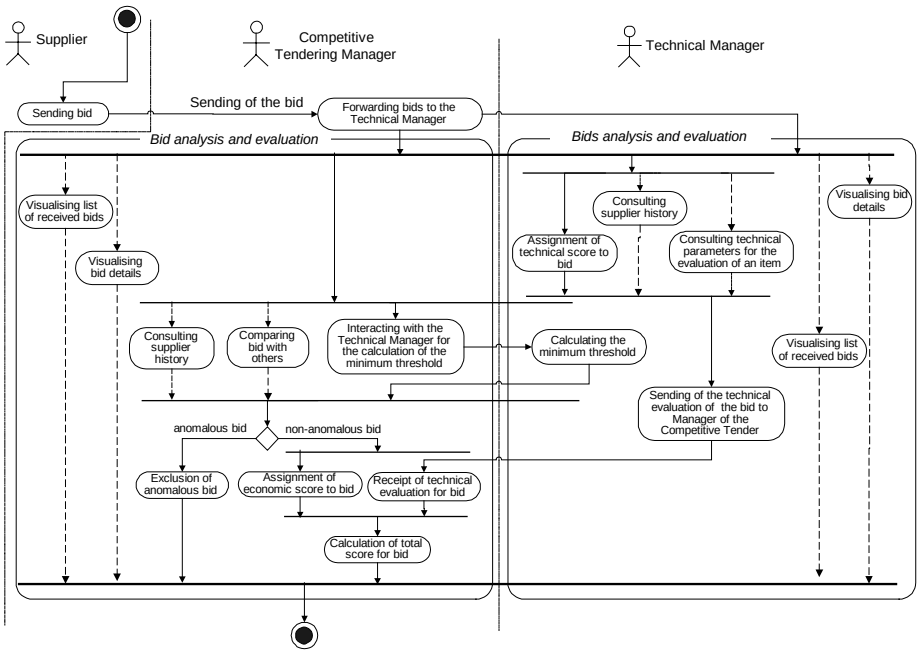


Figure 6 The Execution Model for the activity named 'Bids analysis and evaluation' in the 'Management of the competitive tendering' transaction



6.3 Operation design

In the *Operation Design* phase, for each elementary activity modelled in the Organization Model, the elementary operations that make up its FunctionalSet are identified. They are modelled in terms of pre- and post-conditions, bearing in mind the notion of *context*. The output of the Operation Design is thus the FunctionalSet of the elementary activities.

6.4 Navigation design

The *Navigation Design* activity takes as input the models produced by the transaction design and the Access Structures resulting from the information design. This activity, in contrast to the original specifications of UWA (UWA Consortium, 2001c), goes beyond designing the navigation through the content of the application, and includes the definitions of the relations between Activities of the Transactions Model and the Information objects designed by the Information Model of the application. This design activity is also in charge of defining how contents navigation and activities execution influence each other and the state of ongoing transactions; the two new concepts of Activity Node and Activity Cluster introduced by UWAT+ into the UWA Navigation Model attend to these scopes. Figures 7 and 8 respectively show the Activity Node and the Activity Cluster associated with the activity of 'Assignment of economic score' (compare with the Organization Model in Figure 5 and the Execution Models in Figure 6).

As the Activity Cluster shows, starting from the 'Assignment of economic score' Activity Node, the user can navigate towards the 'Detection of anomalous bids' Activity Node, in order to establish whether the bid is anomalous or not. This navigation does not cause any state transition for the ongoing transaction.

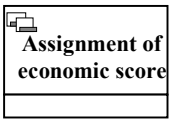
Once the user has established that the bid is not anomalous, s/he can move on to the 'Assignment of economic score' Activity Node, which is intended to support the user in assigning an economic score to the bid being evaluated.

From both these Activity Nodes, it is also possible to navigate towards:

- The Navigation Nodes included in the cluster (this navigation causes the transition of the transaction state in 'Suspended').
- The 'Bid evaluation' Activity Cluster, by clicking on the 'Insert' button action element.

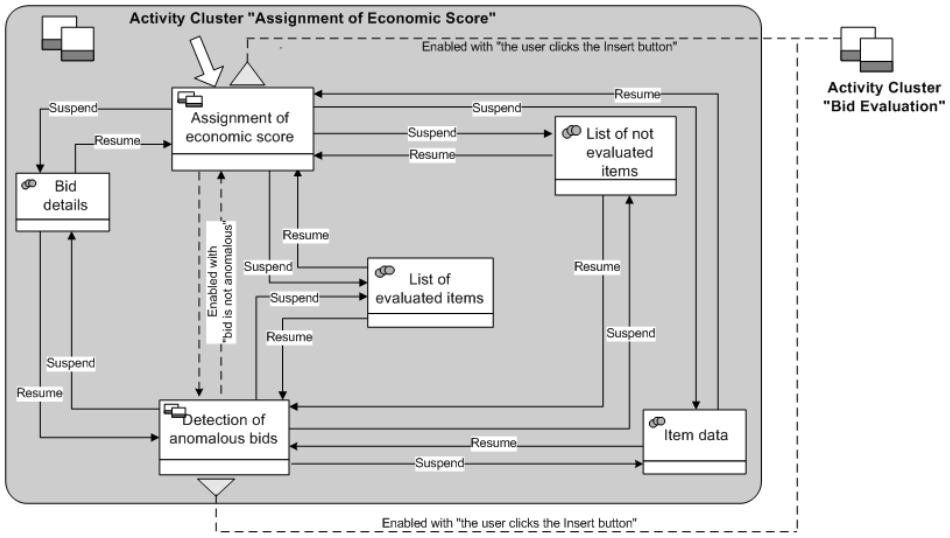
Besides modelling the possible navigation steps the user can follow from a node to the adjacent one, the Activity Cluster also specifies the state transitions of the ongoing transaction associated with each of the navigation and activity links.

Figure 7 The design of the Activity Node associated with the activity 'Assignment of economic score'



Property	Description
Name	Assignment of Economic Score
Source	Assignment of Economic Score elementary activity
Content	The navigation node Bid details is directly involved; the information it contains, relating to the economic score and its motivation, constitute the values of the input variables cache which, if the activity is performed a second time, fills in the fields of the economic score Input form.
Input data	Input variables cache: economic score of the bid and its motivation; if the activity is performed more than once, the values inserted previously are displayed to the user.
Action elements	The *Insert* Button which confirms the user's evaluation and moves the transaction on to the next step.

Figure 8 The design of the Activity Cluster associated with the activity 'Assignment of Economic Score'

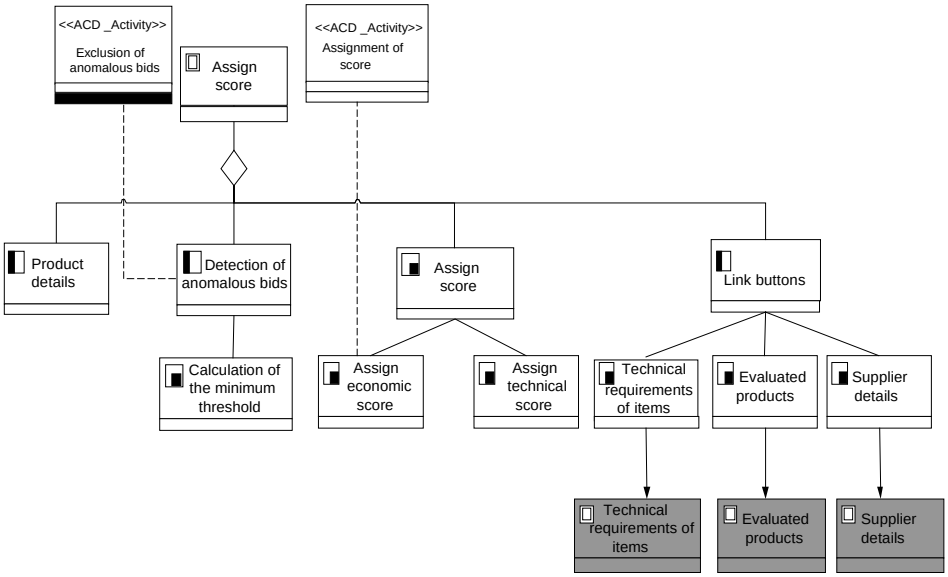


6.5 *Publishing design*

According to the process described by Figure 3, the design of the web application is completed with the UWA Publishing Design activity (UWA Consortium, 2001c). This activity is intended to design the 'final product' of a web application: the pages. During this phase, the nodes and clusters of the Navigation Model are published in publishing units, sections and pages. These three elements make up the output of the Publishing Design phase. Figure 9 shows the publishing model for the 'Assign score' page of the application.

An activity named Customisation Design (omitted in Figure 3) is provided by UWA for designing the ubiquity of the application, that is, the way the application will be adapted depending on the usage context (device, user profile, time, location, *etc.*).

Figure 9 Publishing model for the 'Assign score' web page



7 Discussion

This section discusses UWAT+ main features by describing how it satisfies each of the requirements defined in Section 3. In particular, for each requirement, the UWAT+ primitives that address it are listed.

As described in the comments reported in the following section, the UWAT+ method for designing business processes in web applications fully satisfies the set of identified requirements.

Req1 Identify the component activities of a web transaction, their semantic associations, and the properties/constraints that apply to each of them.

UWAT+ fully addresses this requirement with its Organization Model. As described in Section 5.2, this model enables the designer to represent the set of activities in which a web transaction will be divided and the properties and relations that apply to and between them. Details on the definition of the Organization Model are provided in Section 5.2, while an example of its creation is given in Section 6.

Req2 Describe the possible execution flows (or workflows) of a web transaction.

The UWAT+ Execution Model fully addresses this requirement. A designer adopting UWAT+ will be able to represent all the possible executions flows permitted to the user to complete a web transaction, thus specifying the temporal constraints and the other conditions (*i.e.*, the business rules) that apply to each of the web transaction component activities.

Req3 Define and manage the state of a web transaction.

As described in Section 5.1, UWAT+ explicitly defines a state for a web transaction and for each of its component activities and enables the designer to model how these states evolve because of activity execution or content navigation in the Activity Clusters.

Req4 Specify which activities can be suspended and resumed afterwards during long-lived transactions.

UWAT+ enables the designer to specify if an activity can be suspended both in the Organization Model (by means of the Suspendability property of the PropertySet) and in the Navigation model (by representing the transitions that will suspend the transaction in the Activity Clusters).

Req5 Describe the way two or more types of users involved in a web transaction collaborate in its execution.

By using swimlanes in the Execution Model, the designer using UWAT+ to design business processes in web applications will be able to design how different types of user collaborate in the process being responsible for executing which activities.

Req6 Specify the way content navigation and operation execution affect each other and the state of an ongoing web transaction.

By modelling the UWAT+ Activity Clusters associated with an activity of a web transaction, the designer defines:

- which navigation steps are permitted to the user when in the considered Activity Node
- from which Navigation or Activity Node the user may get into the considered Activity Node
- how the state of the web transaction changes when the considered activity is executed or the user navigates towards a different Navigation Node or Activity Node.

State transitions are indicated onto Navigation and Activity Links.

Req7 Define which contents will be provided to the user in order to support the execution of a particular activity.

This requirement is specifically addressed by the UWAT+ Activity Node design primitive. The Content attribute of an Activity Node together with the UWA Navigation node notion, enables the designer to specify which contents will be provided to the user while being in the page that enables him to execute an activity.

Req8 Define which information objects are affected by the execution of the activity and how.

Impacts of the execution of an activity on the application information objects (such as Entities and Semantic Associations defined in the application Hyperbase, or Access Structures and Nodes defined in the Navigation Model)

are modelled by the UWAT+ Activity Nodes. Where necessary, in addition to UWAT+ Activity Nodes, the UWA Operation Design (UWA Consortium, 2001c) can be used to define a more detailed and formal design of this aspect of web transactions.

Req9 Describe the way an activity will be customised depending on the state of the ongoing transaction.

In UWAT+, different Activity Nodes and Activity Clusters may be designed for a certain activity, or the customisation rules that apply to the default Activity Node and Cluster associated with a certain activity may be defined. In both cases, the aim is to specify how a certain activity will be customised depending on the state of the ongoing transaction or the execution context. Refer to Section 5 for further details.

Req10 Describe the way an activity will be customised depending on the context of execution.

For this requirement the same considerations reported for Req9 apply.

8 Conclusion and future work

This paper addressed the problem of designing business processes in web applications. It has been widely discussed that the addition of business processes to web applications gives rise to challenges and risks that require a suitable design methodology in order to improve usability and avoid undesired behaviours. To cope with these needs, a number of models and methods originally proposed for the design of informational hypermedia applications, such as UWA, UWE, OOHDM, WSDM and WebML have been extended to provide the designer with a way of addressing the design of business transactions.

Based on experience in designing a number of industrial business web applications by using the UWA design framework and in analysing and comparing each of the above-mentioned design methods, a representative list of requirements to be satisfied by a suitable method for designing business process in web applications was elaborated. By identifying room for improvements in the UWA Transaction Design Model and borrowing solutions adopted by the analysed design methods, an extension of the model, called UWAT+, was proposed that fully satisfies the set of identified requirements.

By means of the new design primitives added to the UWA Navigation model, namely the Activity Node and Activity Cluster, and thanks to the other extension introduced in the original UWA Transaction Design model, such as the state of a transaction, the presented design method satisfies all the set of requirements. UWAT+ enables the designer to effectively capture and model the transactions of a web application according to the user's perspective, to clearly represent the way transaction execution and content navigation interact with each other, and to define which information objects are needed by the user when executing a given activity and how a component activity will be customised depending on the execution context and state.

The paper illustrated the approach with a comprehensive example in the domain of public tendering. Other real-world case studies have been successfully developed thus validating the approach and its effectiveness.

Ongoing research regards the development of tools to support the design method by extending existing UWA design tools and the development of an approach to reengineering of legacy applications to the web.

Future work

There are a number of possible avenues of future work suggested by the results presented in this paper. These areas include developing tools to support the design methodology, applying the methodology to the reengineering of legacy web applications, mapping UWAT+ design models into application built around services, and integrating the design method with other modelling approaches in the domain of business processes or workflow specification, such as BPML (ebPML.org, 2002) and WF-XML (WfMC.org, 2004).

As presented in this paper, the approach is descriptive in nature. There is a place for the introduction of tools and automated support. There are UWA design tools and development aids that could form the basis for this effort.

The large number of existing legacy applications implementing business processes and the interest in migrating these applications onto a web-based platform constitute the underlying motivation for the definition of a reengineering approach based on UWAT+. Preliminary work in this area has already begun, with promising initial results (Distante *et al.*, 2006a–b).

Incorporating business process issues more deeply into the design models is something that is seen as beneficial. Such a holistic approach might prove more suitable to adoption by end users, since for most enterprises, process and content are intimately related to one another.

The UWAT+ design model may also be helpful in reverse engineering tasks, to analyse, document, comprehend and evolve the design of web transactions in existing web applications, often suffering from lack in documentation. Early results from work done in this direction have shown to be promising (Distante *et al.*, 2004a–c; Tilley *et al.*, 2005).

Service-oriented architecture is currently enjoying considerable attention. It offers the possibility of engineering large-scale systems as a set of interacting on-demand services, providing the customer with a solution that is much more business-oriented and less focused on the underlying technology. Integrating UWAT+ design models into such service-based systems could prove beneficial.

Finally, integrating the UWAT+ design approach with other modelling approaches, such as those used in business process modelling or workflow analysis, is an intriguing proposition. Since UWAT+ relies on standards (*e.g.*, UML), and many other modelling approaches also use (emerging) standards (*e.g.*, BPML), the combination of the two could again foster the adoption of the overall approach.

References

Atzeni, P. and Parente, A. (2001) 'Specification of web application with ADM-2. Araneus project, Università Roma Tre', *Proceedings of the 1.er Workshop Internacional de Tecnologia Software Orientada a Ambiente Web*, Valencia, Spain.

Atzeni, P., Mecca, G. and Merialdo, P. (1997) 'To weave the web', *Proceedings of the 23rd International Conference on Very Large Databases (VLDB'97)*, Athens, Greece, 25–29 August.

Baresi, L., Denaro, G., Mainetti, L. and Paolini, P. (2002) 'Assertions to better specify the Amazon Bug', *Proceedings of the 14th International Conference of Software Engineering and Knowledge Engineering (SEKE 2002)*, Ischia, Italy, 15–19 July.

Brambilla, M. (2003) 'Extending hypertext conceptual models with process-oriented primitives', *Proceedings of International Conference on Conceptual Modeling (ER 2003)*, October, Chicago, Illinois, pp.246–262.

Brambilla, M., Ceri, S., Fraternali, P. and Manolescu, I. (2006) 'Process modeling in web applications', *ACM Transactions on Software Engineering and Methodology (TOSEM)*, in print.

Ceri, S., Fraternali, P. and Bongio, A. (2000) 'Web Modeling Language (WebML): a modeling language for designing web sites', *Computer Networks*, Vol. 33, Nos. 1–6, pp.137–157.

Chrysanthis, P. and Ramamritham, K. (1994) 'Synthesis of extended transaction models using ACTA', *ACM Transactions on Database Systems*, September, Vol. 19, No. 3, pp.450–491.

De Troyer, O. and Casteleyn, S. (2003) 'Modeling complex process for web application with WSDM', *Proceedings of the 3rd International Workshop on Web Oriented Software Technology (IWWOST 2003)*, Oviedo, Spain, 15 July

De Troyer, O. and Leune, C.J. (1998) 'WSDM: a user centered design method for web sites, computer and ISDN systems', *Proceedings of the 7th International World Wide Web Conference*, Elsevier, pp.85–94.

Distante, D. (2004) 'Reengineering legacy applications and web transactions: an extended version of the UWA transaction design model', *Ph.D. Dissertation*, University of Lecce, Italy, June.

Distante, D. and Tilley, S. (2005) 'Conceptual modeling of web application transactions: towards a revised and extended version of the UWA transaction design model', *Proceedings of the 11th International Multi-Media Modelling Conference (MMM 2005)*, 12–14 January, Melbourne, Australia, Los Alamitos, CA: IEEE Computer Society Press.

Distante, D., Parveen, T. and Tilley, S. (2004a) 'Towards a technique for reverse engineering web transactions from a user's perspective', *Proceedings of the 12th IEEE International Workshop on Program Comprehension (IWPC 2004)*, Bari, Italy, Los Alamitos, CA: IEEE CS Press, 24–26 June.

Distante, D., Tilley, S. and Huang, S. (2004b) 'Documenting software systems with views IV: documenting web transaction design with UWAT+', *Proceedings of the 22nd International Conference on Design of Communication (SIGDOC 2004)*, Memphis, TN, New York, NY: ACM Press, 10–13 October.

Distante, D., Tilley, S. and Huang, S. (2004c) 'Web site evolution via process restructuring from a user's perspective', *Proceedings of the 6th IEEE International Workshop on Web Site Evolution (WSE 2004)*, Chicago, IL, Los Alamitos, CA: IEEE Computer Society Press, 11–17 September.

Distante, D., Tilley, S. and Canfora, G. (2006a) 'Towards a holistic approach to redesigning legacy applications for the web with UWA and UWAT+', *Proceedings of the 10th European Conference on Software Maintenance and Reengineering (CSMR 2006)*, Bari, Italy, 22–24 March, pp.293–297.

Distante, D., Tilley, S., Canfora, G. and Huang, S. (2006b) 'Redesigning legacy applications for the web with UWAT+: a case study', *Proceedings of the 28th International Conference on Software Engineering (ICSE 2006)*, Shanghai, China, 20–28 May.

ebPML.org (2002) *BPML 1.0 – Business Process Modeling Language*, http://www.ebpml.org/bpml_1_0_june_02.htmWF-XML.

Elmasri, R. and Navathe, S. (1998) *Fundamentals of Data Base Systems*, 4th ed., Benjamin-Cummings.

Fischetto, G. (2004) 'Designing web transactions: theoretical and practical aspects', *M.Sc. Thesis in Computer Science Engineering*, University of Lecce, Italy, February (in Italian).

Gioldasis, N. and Christodoulakis, S. (2002) 'UTML: Unified Transaction Modeling Language', *Proceedings of the Third International Conference on Web Information Systems Engineering (WISE'02)*, Grand Hyatt, Singapore, 12–14 December.

IDEF-0 (1993) *Integrated-Computer-Aided Manufacturing Definition, Activity Model, Integration Definition for Function Modeling*, 31 December, NIST FIPS Pub 183.

Koch, N., Kraus, A., Cachero, C. and Meliá, S. (2003) 'Modeling web business processes with OO-H and UWE', *Proceedings of the 3rd International Workshop on Web Oriented Software Technology (IWWOST 2003)*, Oviedo, Spain, 15 July.

Object Management Group (OMG) (2004) *Unified Language Modeling Specification*, Version 2.0, www.omg.org.

Patern, F., Mancini, C. and Meniconi, S. (1997) 'ConcurTaskTree: a diagrammatic notation for specifying task models', in S. Howard, J. Hammond and G. Lindgaard (Eds.) *Proceedings of The 6th IFIP International Conference on Human-Computer Interaction (INTERACT '97)*, Sydney, Australia, 13–18 July.

Rossi, G., Schmid, H. and Lyardet, F. (2003) 'Engineering business processes in web applications: modeling and navigation issues', *Proceedings of the 3rd International Workshop on Web Oriented Software Technology (IWWOST 2003)*, Oviedo, Spain, 15 July.

Schmid, H. and Rossi, G. (2004) 'Modeling and designing processes in e-commerce applications', *IEEE Internet Computing*, January/February.

Schwabe, D., de Almeida Pontes, R. and Moura, I. (1999) 'OOHDM-WEB: an environment for implementation of hypermedia applications', *WWW, SIGWEB Newsletter*, June, Vol. 8, No. 2.

Tilley, S., Distante, D. and Huang, S. (2005) 'Design recovery of web application transactions', in H. Yang (Ed.) *Advances in Software Evolution with UML and XML*, Hershey, PA: Idea Group Publishing, May.

UWA Consortium (2001a) *Deliverable D2: General Definition of the UWA Framework*.

UWA Consortium (2001b) *Deliverable D6: Requirements Elicitation: Model, Notation and Tool Architecture*.

UWA Consortium (2001c) *Deliverable D7: Hypermedia and Operation Design: Model and Tool Architecture*.

UWA Consortium (2001d) *Deliverable D8: Transaction Design*.

UWA Consortium (2001e) *Deliverable D9: Customization Design*.

UWA Consortium (2002) 'Ubiquitous web applications', *Proceedings of The eBusiness and eWork Conference (e2002)*, 16–18 October, Prague, Czech Republic.

WfMC (2006) *Workflow Management Coalition*, http://www.wfmc.org.

WfMC.org (2004) *Wf-XML 2.0 – XML Based Protocol for Run-Time Integration of Process Engines*, http://www.wfmc.org/standards/docs/WfXML20-200410c.pdf.