# Combining web engineering methods to cover lifecycle

**K Wakil[1, 2, 3*], D N A Jawawi[1]**

*[1]Department of Software Engineering, Faculty of Computing, UTM, Malaysia*
*[2]Sulaimani Polytechnic University, Iraq*
*[3]University of Human Development, Iraq*

*[*]Corresponding author: karzanwakil@gmail.com*

*Received 6 December 2016, www.cmnt.lv*

**Abstract**

Web applications have rapidly evolved in the last decade, whilst web engineering methods have been lacking in the process development Web applications. One of the issues in web engineering methods is that no single web engineering method provides adequate coverage for the whole life cycle, because the web engineering methods are divided into three phases, which are; requirements, analysis/design, and implementation. Therefore, each method designed to special concern. It is obvious that we need to design a new method to cover the whole lifecycle to solve this issue. In this paper, we propose a framework for the new web engineering method through a combination of three methods comprising: Navigational Development Techniques (NDT) method for requirements phase; UML-Based Web Engineering (UWE) for analysis/design phase; and Interaction Flow Modeling Language (IFML) for the implementation phase. NDT and UWE are the most representative methods to develop web applications; while IFML is the newest method that focused on design and implementation. Our framework for the new method can support a whole lifecycle. Moreover, this method is more usable from developers.

*Keywords*

Web engineering,
methods,
combination,
lifecycle,
IFML

## 1 Introduction

Model Driven Web Engineering (MDWE) methods such as WebML [1], IFML [2], UWE [3] or OOHDM [4] during the last years have turn out to be established solutions for developing Web applications. These methods use Model Driven Development (MDD) ideas to attract high-level Web applications concepts into models and apply these models to derive applications automatically. The process of classic MDWE development consists of three phases [5]: (1) building a domain model, (2) defining a hypertext model and (3) defining the application's look and feel. A set of models is the outcome of the process that can create the last Web application using code generation. Moreover, the Model Driven Architecture (MDA) based development process establishes four phases of the development life cycle: analysis; platform independent design, where a Platform Independent Model (PIM) is built; Platform Specific Design, where a platform specific model (PSM) is built; and implementation [6].

Several issues within web engineering methods do exist. One of them is that there is no single method that covers the entire development life cycle in depth, and each method bears has its own particular strengths [7-8] as illustrated in Figure 1. As it is established in [9], the majority of the methods that are intended just for the hypermedia systems design partly cover the hypermedia systems life cycle and are further concentrated on the systems design. The web engineering community and several research groups are geared towards sustainable solutions to such variations, with some being solved by merging two methods like RUX-Method and UWE method to support Rich Internet Applications (RIA) [10], while the solution of others was obtained through enhanced methods like UWE metamodels in establishing novel modules of websites [11] although could never have all the issues completely solved.

To solve this issue, there is a necessity to design a new method. There are three ways to define a new web engineering method which include: extension existing methods; combine existing methods, and define new method. Nevertheless, in [9] the researcher recommended for new method to cover lifecycle with combination by employing common model, but in this paper we propose a framework for defining a new web engineering method through merging three approaches by metamodels with adopt strong models.

In order to design the framework for the new method, we selected the most representative methods [12] as follows: NDT method for the requirements phase; UWE for the analysis/design phase; and the newest method, IFML instead of WebML, for the implementation phase. For define a new method we propose a new framework, in this framework we use strong model in each method to cover lifecycle, here we use requirements model from NDT, design model from UWE, and Implementation model from IFML. The new method will be more usable and interoperable method to develop web applications.

The paper is organized as follows: Section 2 explains the background work undertaken for the web engineering methods during the lifecycle, and some combination between web engineering methods. In section 3, we analyze the most representative web engineering methods that used to support lifecycle. In Section 4, we propose a framework to design a new method to cover lifecycle. In the last section, we present some concluding remarks and suggestions for future research.

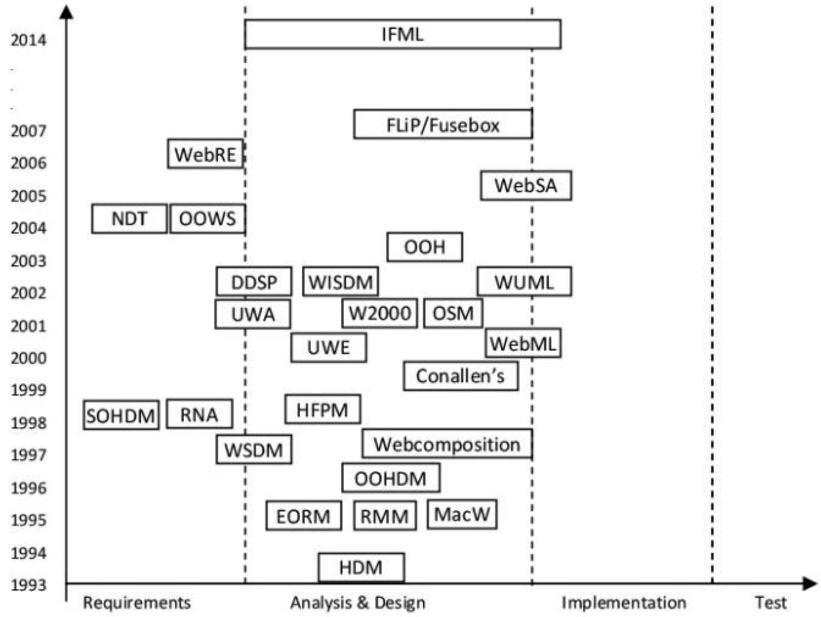


FIGURE 1 the evolution and coverage the best-known web development [7]

## 2 Background

Several MDWE methods were offered and they solved the complexities of methods to development web applications during the past years. However, they as well presented some restrictions and one of the limitations is that there is no single method to cover lifecycle deeply. Within this part, we evaluate several literature works to resolve lifecycle and combination within MDWE methods.

Numerous approaches, methods, and processes have been proposed in the educational and expert literature over the previous years to handle special features of Web development, and expert literature in MDWE [13]. Intricate interfaces, navigation, complex maintenance, safety features and unidentified remote users are among the serious difficulties pertinent to Web-based system improvement. Nevertheless, they resolved the challenges although they as well present some limits. One among the limits is lack in cover lifecycle [6, 14-15]. Lang and Fitzgerald [16] offer an all-inclusive record of overfly techniques and methods for Web hypermedia systems expansion. An explanation and comparative study of the better recognized of these Web development methods can be found in [17].

A significant perception in [6] as shown in Figure 1 is the assorted coverage by methods of the development phases. Within the Figure 1, every method is positioned in the phase where its major concentration lies. Therefore, even though the UWA Project [18] or WebML [19] offer some thought to necessities description and implementation, they largely highlight the examination and design phase. As can be viewed, most of Web development methods are focused on analyze and design phase, with perceptibly less concentration on the other life cycle phases.

Several union web engineering methods exist with every combination done for resolving a diverse difficulty such as; Preciado, J. C. et al. in [10] united RUX-Method and UWE to support RIA. They recommend a model-driven method to RIA development by uniting the UWE method for data and business logic sampling through the RUX-Method for the consumer interface sampling of RIAs.

Preciado et. al [20] employ RUX-Model [21], a MDWE Method for the systematic adaptation of RIAs UIs over existing HTML-based Web applications based on models in order to give them multimedia support, giving more efficient, interactive and instinctive user experiences. Amongst the phases of transformation proposed in RUX-Model, they have concentrated on the description of the connection procedure having the Web model being modified. This phase is decisive in the procedure because of it being the lone element of RUX-model that relies on the Web model chosen [22].

In [23] the researchers offered the Method Association Approach (MAA) that chooses and constructs appropriate methods from five model-driven web modeling methods. The MAA forms modeling approaches in definite web application domains for uses in diverse phases of their life sequence. The MAA places existing model-driven methods using metamodel ideas against key aspects of a particular web application continuum. Through the MAA, a design approach is built that flawlessly adjusts to web application aspects, and utilizes confirmed concepts of web design. The method has been confirmed using specialist corroboration and analysis of the two cases. In another study [24] the authors submitted a method and device support to construct web information networks that combine the employ of Scrum methods and Model-Driven Engineering (MDE). Such method and device permit performing fast design and corroboration of pre archetype models.

Daniel and Pozzi proposed a framework for the design and development of adaptive Web applications. The framework leverages on the integration of two well established methods: a conceptual model, complemented with a CASE tool for automatic code generation, and a language for expressing ECA rules, supported by an engine for rule execution. Such integration leads to a versatile and flexible adaptivity environment, whose advantage is twofold: on one hand, conceptual modeling and automatic code generation support an efficient development process; on the other hand a detached rule engine allows us to widen the set of adaptivity requirements that can be handled and to overcome some limitations of current modeling approaches [25].

Huang et. al in the ref [26] presented an expanded lifecycle process model for the development of Web-based applications in SMEs. It consists of three sets of processes, meaning requirement processes, development processes, and evolution processes. Particularly, the post-delivery evolution processes are important to SMEs to develop and maintain quality Web applications with limited resources and time.

In the excellent work Rivero et. al, used User Interface prototypes (usually known as mockups) as a way to start the modeling process in the context of a mixed agile-MDWE process. To assist this process, we defined a lightweight metamodel that allows modeling features over mockups, interacting with end-users and generating MDWE models. Then, we conducted a statistical evaluation of both approaches (traditional vs. mockup-based modeling) [27].

The best combination to cover lifecycle presented in [8], in this paper suggested three web Engineering methods that are NDT,UWE, and WEML to cover lifecycle as shown in Figure 2, however this idea is best idea but for implementation very hard because used common model and need to new transformation model, moreover no tool support for implementation this idea.

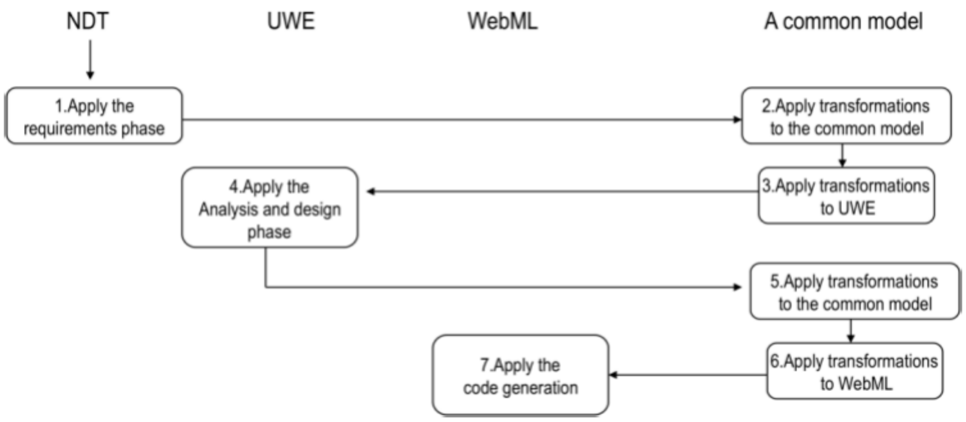


FIGURE 2 Use common metamodels to make approaches compatible[8]

One particular aspect of web engineering that remains problematic is the lack of integrated toolsets to support development methods and approaches, a long-standing difficulty alluded to some years ago in [28]. Because of the frequent changes in Web systems and the imperative to release fully functional upgrades quickly and often, Web development methods must be highly agile. The use of CASE tools that provide automated processes and enable rapid development/re-factoring is therefore necessary. In recent years, methods such as UWE, which offers a tool named MagicUWE [29], and WebML, which is supported by the WebRatio tool [30], have been greatly welcomed. Nevertheless, for CASE tools to be interoperable and interchangeable between and across Web development methods, it is essential that there must be a mechanism to facilitate the transformation and consistent integration of semantic metamodels. In this regard, MDWE offers much promise because it potentially enables Web developers to mix-and-match method fragments taken from different approaches and combine them into a tailored hybrid which is customized to the needs of a particular development project. This paper offers a critical view about this possibility by analyzing if approaches can be easily integrated or extended with new approaches.

## 3 Analyzing Web Engineering Methods

In this section we analyze the majority preventive methods that used to resolve lifecycle issue. The methods include NDT Method for Requirements phase; UWE for analyze and design phase and IFML for implementation phase, in the following we explained one by one.

### 3.1 NDT

NDT [31] is an MDWE methodological approach mainly focused on requirements and analysis. NDT describes a collection of PIM and CIM models and the set of revolutions by QVT to coin PIM from CIM. Similar to it happens in other methods, these metamodels are described by employing class diagrams. The necessities metamodel of NDT is an expansion of WebRE that comprises new ideas depending on the WebRE method. Moreover, it comprises two metamodels, the navigational and the content for the PIM level. The latter is the UML metamodel for class illustrations and the former is the metamodel for the UWE. One of the largely significant benefits of this method is its device support. A set of devices known as NDT-Suite, fabricated of four devices supports the MDE development process of NDT (this device-set can be got at [32]). Every NDT metamodel has a particular profile that is executed in Enterprise Architect [33]. The NDT method has modified the interface of this device through a set of device boxes having direct contact to every artifact of the method. This atmosphere is known as NDT-Profile. Besides, NDT-Suite comprises six other devices which are NDT-Driver, NDT-Report, NDT-Quality, NDT-Glossary, NDT-Prototypes, and NDT-Checker.

In Figure 3, the NDT development process can be defined as a bottom-up process. The process of development is concentrated on an extremely comprehensive necessities definition, directed by objectives that cover three sub-phases: necessities capture, requirements description, and requirements corroboration. NDT simply covers the initial phases within the life cycle. Moreover, it is essential to highlight that workflows within NDT that shift from necessities to analysis are methodical. These workflows are described by means of the MDE paradigm. The need of offering a systematic procedure so as to create Web design models has been noticed by numerous investigation groups. These workflows might yet be mechanical if the development group utilizes its related device of NDT, the NDT-Tool.

When necessities are confirmed, the NDT procedure goes on by describing three models:

- The content model that is a class illustration. It articulates the static outlook of the system.
- The navigational model that reveals how consumers can navigate through the method.
- The abstract interface model that displays the theoretical interface of the structure.

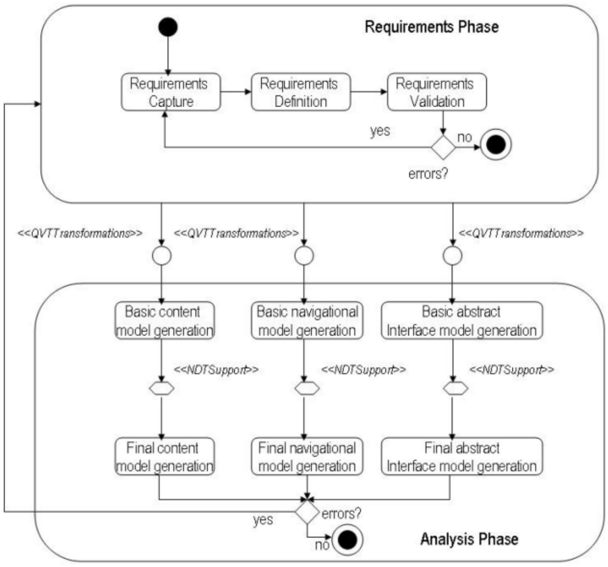


FIGURE 3 NDT development process [31]

## 3.2 UWE

UWE came up by 1998. The method was developed by the Web Engineering Group from the Ludwig-Maximilians-Universität München [34-35]. UWE is a software engineering approach based on UML [36] it uses the UML standard notation as much as possible and defines a UML profile to specify the peculiarities that introduce web applications. The major benefit of being UML compliance is that any CASE tool that supports the UML notation can be used to produce the UWE models [37].

UWE a method based on standards. UWE focused on visual modelling together with systematic design and automatic generation also UWE is a software engineering approach for the web applications whose objective is to cover the entire life cycle of Web application development.

UWE is said to be a small extension of UML, which provides UML profile for the web domain. UWE also provides some tolls which can which can be a lot helpful in model designing and consistency checks and automatic creation of web applications. There are two UWE plug-ins called ArgoUWE and MagicUWE which can support notation of UWE portfolio and design is also supported with the help of transformation [38].

UWE metamodel is a design considered as the conservative extension of the UML metamodel, in other words we can say that the modelling elements of a UML metamodel are inherited from the UML metamodel and they are not modified by adding new features or additions to the modelling elements class. Any additional features or relationship if we want to implement then they can be specified in different metamodel modelling element and then define OCL restrictions on additional static semantics and it is equivalent to well formatted rules in the UML specs. We can have benefit from the metamodelling tools that are depending upon equivalent XML Interchange (XMI) format by keeping them compatible with the Meta Object Facility (MOF) interchange metamodel [39].

The UWE Metamodel can be customized on the basis of a profile by mapping it to a UML profile.UWE metamodel for web applications can be created by using generic UML case tools and UML profiles or their extension i.e. typecast, objects those are tagged and OCL restrictions. CASE tools can be used to maintain UWE method but that actually depends on the technical feasibility. If we are bonded to use UML version then we also need to take the consideration of problems in its specification. UML metamodel is included in metamodel architecture for OMG for example, considering a metamodel arch with different levels in it, then a modelling element at level 2 is not an instance of exactly one element at level 3. This is also called as a ‘loose metamodelling problem’, which can be taken care of in new versions of UML [39-40] .

UWE approach proposes to build a set of CIMs, PIMs, and PSMs as results of the analysis, design and implementation phases of the model-driven process. The aim of the analysis phase is to gather a stable set of requirements. The functional requirements are captured by means of the requirements model. The requirements model comprises specialized use cases and a class model for the Web application. The design phase consists of constructing a series of models for the content, navigation, process, presentation and adaptivity aspects at a platform independent level. Transformations implement the systematic construction of dependent models by generating default models, which then can be refined by the designer, as shown in Figure 4 [41].

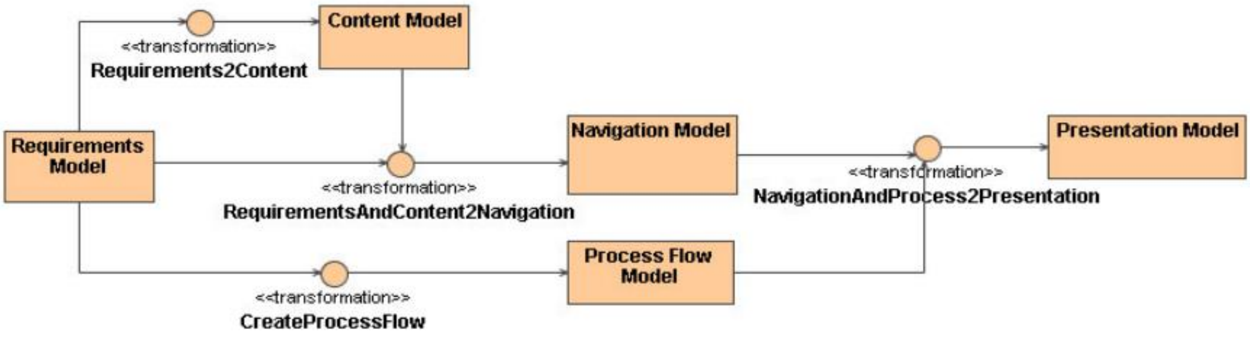


FIGURE 4 UWE core process [41]

3.3 IFML

IFML[2] supports the platform-independent description of graphical user interfaces for applications deployed or accessed on systems such as desktop computers, laptops, PDAs, mobile phones, and tablets. The main focus is on the structure and behavior of the application as perceived by the end user. The modeling language also incorporates references to the data and business logic that influence the user's experience. This is achieved by referencing the domain model objects that provide the content displayed in the interface and the actions that can be triggered by interacting with the interface.

The development of applications defined by interactivity is normally handled with agile techniques, which navigate diverse phases of "problem identification/design modification/implementation." The iteration of the creation method derives a partial version or a prototype of the system. Such an augmentable lifecycle is predominantly suitable for contemporary web and mobile uses, with the need of being installed swiftly and alter frequently throughout their lifetime to adjust to user prerequisites. Figure 5 offers a probable structural creation process hence positioning IFML within the activity flow.

**Domain Modeling** systematizes the key information objects established during conditions delineation into a broad and articulate setting model. Domain modeling delineates the key data sets established during conditions requirement into a domain model, normally a (characteristically visual) depiction of the necessary objects, their qualities and relationships.

**Front-End Modeling** plots the data manipulation and information conveyance functionality proposed by the requirements application conditions into front-end model. The operation of front-end modeling is at the conceptual angle, with IFML coming into play. The developer is at the liberty of utilizing IFML in the specification of front-end organization in a single or several top-level view containers, the internal formation of every view container regarding sub-containers, the constituents forming each view container's content, the events depicted by the components and vie containers, as well as how such events set off business events and revise the interface.

**Business Logic Modeling** delineates the business objects and the techniques needed to sustain the established use cases. UML dynamic and static figures are usually used in highlighting the objects interface as well as messages flow. Process-adjusted details (like UML functionality and sequence charts, BPMN process models, and BPEL service orchestrations) offer an efficient method of signifying the workflow across services and objects. The services highlighted in the business logic plan can be oriented in the front-end model to signify the operations to be set off through interface interaction. Being interdependent in nature, front-end, data, and business-logic structure events are performed in an iterative manner. The preference category of Figure 5 is simply indicative. Within some companies, the responsibility could commence at the structure of the front-end while the actions and data objects could be established at a later phase though analysis of the published information as well as the requested operations towards sustaining the interactions.

Architectural structure is the technique of delineating the network, hardware as well as the software elements that compose the architecture whereby the application offers its services to the users. The objective of the architectural structure is to establish the mixture of these components that adequately achieves the application needs as regards to scalability, efficiency, accessibility, security, and all together adhering to the economic and technical project limitations.

**Implementation** entails the approach of creating the software modules that convert the business logic, data as well as interface design into an application functioning on the opted design. Implementation of data situates the domain model onto a single or several data sources by merging the conceptual-level aspects with the formations of logical data (such as relationships and aspects to relational tables). The execution of business logic generates the software components required to sustain the identified use cases. The execution of individual entities may gain from the adoption of software designs, which systematize the manner in which fine-grain elements are devised and merged into a wider and highly reusable operational units and equally provide for nonfunctional needs like scalability, accessibility, security and competence. Translation of abstract-level ViewComponents and ViewContainers into the apposite aspects within the considered execution plan is done courtesy of interface accomplishment. It is possible for the ViewContainers and business objects to interoperate either in the server or client layer.

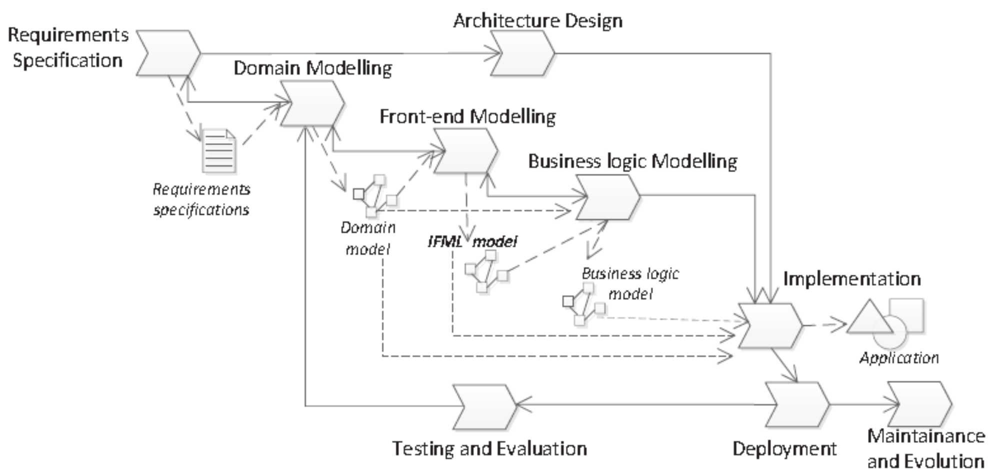


FIGURE 5 the role of IFML in the development process of an interactive application

**Testing and Evaluation** confirms the consistency of the installed application concerning the nonfunctional and functional conditions. The key important aspects for interactive model testing include:

- Functional Trialing: verification of the application behavior regarding the functional needs. Functional testing is disintegrated into classical events of module examination, system testing and integration testing.
- Usability Assessment: the nonfunctional prerequisites of accessibility, communication efficiency, and observance to merged usability values are confirmed against the generated front end.
- Performance Assessment: the application's response time and throughput ought to be examined in peak and average workload provisions. There is the need to monitor and examine the insufficient service levels, the usability design, so as to establish and get rid of bottlenecks.

## 4 Proposal of new web engineering method to cover lifecycle

In this section, we define a framework for a new web engineering method which can satisfactorily cover the whole life cycle. As mentioned in previous sections, we are using the most representative methods, namely, NDT, UWE, and IFML. Moreover, these methods are used to achieve better comparison and implementation. Each method has a particular strength in the process development of a lifecycle, as follows: NDT is a method focused on requirements; UWE is focused on analysis and design; and IFML is focused on Implementation. Already existing were some ideas covering life cycle, but these were difficult to implement. In this section, we define a new method through which to borrow models from one method to another. In Figure 6, we present answers to questions of "how, which, and where" a model can borrow attributes from another model.

- In Level A, we present the methods for particular strengths in a lifecycle. For the requirements phase, we have selected the NDT method; while for the analysis and design phase, we have selected UWE. For the implementation phase, we have selected IFML.
- In Level B, we present the important models which are used for the development of web applications. As mentioned in Section 3, NDT has three models which are: requirements capture; requirements definition; and requirements validation respectively. UWE has five models comprising: requirements model; content model; navigation model; process flow model; and presentation model. IFML has three models which are: Domain Model; IFML model; and Business Logic model. These also support code generation.
- Level C, presented the case tool by which to support methods, namely: NDT supported by NDT-Suite; UWE developed by ArgoUWE and MagicUWE; and IFML developed by WebRatiotool.
- Level E, presents a new web engineering method by combining strengths of each of the models NDT, UWE, and IFML, and implementing this inside IFML. In addition, we use WebRatio tool for the development of a new web engineering method.
- In Level D, we represented the strengths of each of the models, in particular, the model from NDT.

Furthermore, we recognized the need for transformation models for moving from one phase to another such as, CIM and PIM respectively.

As shown in Figure 6, we defined a new framework for a web engineering method; however, we could not implement this due to some challenges that were encountered. These challenges comprised: transformation models between levels became a significant challenge for implementation; a considerably long time was needed for implementation; the very complex work required was best suited for group work rather than an individual researcher; and there was a necessity to improve tool support. For these reasons, we could not apply a case study by our framework.

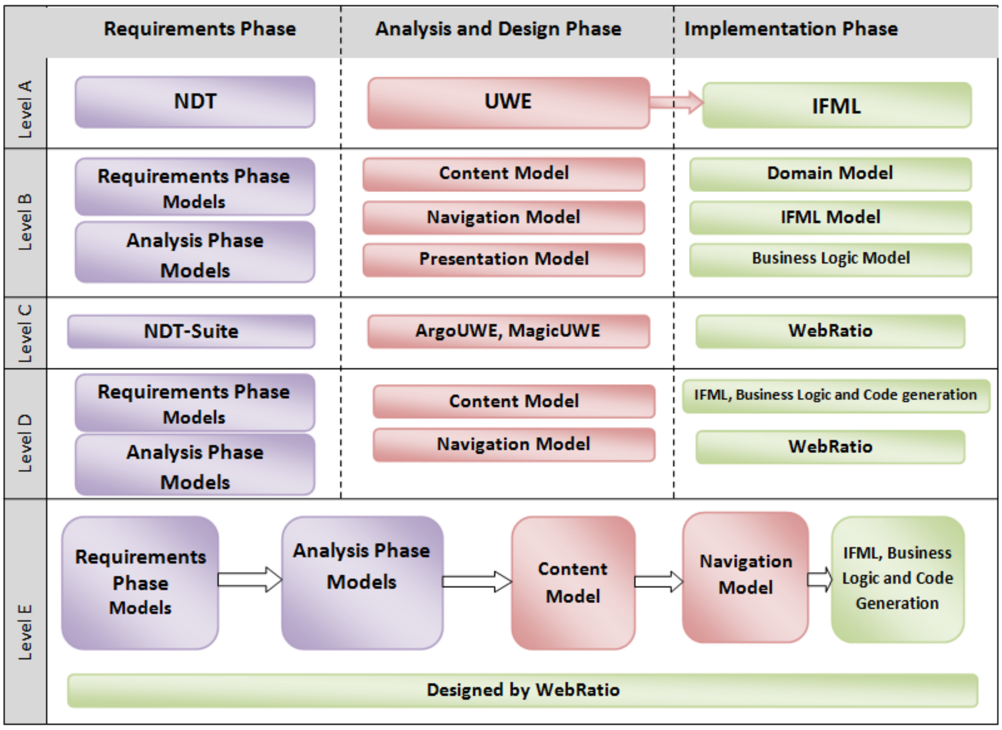


FIGURE 6 New Web Engineering method to Cover Lifecycle

## 5 Conclusion and future work

In this paper, we have defined a framework for a new web engineering method to cover lifecycle by using three methods that have particular strengths in the web engineering lifecycle. We propose a new web engineering method through a combination of three methods, namely: NDT method for the requirements phase; UWE for the analysis/design phase; and IFML for the implementation phase. Our method can support the whole lifecycle; it is also a compatible and interoperable method with which to support web development. Moreover, this method is more usable for implementation. Our recommendations for researchers are to implement this method by existing tools or extension of an existing tool. A new case tool can also be created; moreover, it can import more models to improve our method from other methods for the development of web applications. In addition, new models can be defined, including: adaptivity model; security model and so on. A suggestion for future work is the addition of model transformation and implementation case study by WebRatio Tool.

## References

[1] Ceri S, et al. 2002 Conceptual modeling of data-intensive Web applications *Internet Computing, IEEE* **6** 20-30

[2] Brambilla M, Fraternali P 2014 Interaction flow modeling language: Model-driven UI engineering of web and mobile apps with IFML: Morgan Kaufmann

[3] Koch N, et al. 2008 UML-based web engineering *Web Engineering: Modelling and Implementing Web Applications, ed: Springer* 157-91

[4] Rossi G, Schwabe D 2008 Modeling and implementing web applications with OOHDM *Web engineering: Modelling and implementing Web applications, ed: Springer* 109-55

[5] Wimmer M, et al. 2007 On the integration of web modeling languages: Preliminary results and future challenges *Workshop on Model-driven Web Engineering (MDWE), held in conjunction with ICWE, Como, Italy*

[6] Hincapié Londoño J A, Duitama J F 2012 *Model-driven web engineering methods: a literature review* Revista Facultad de Ingeniería Universidad de Antioquia 69-81

[7] Jawawi K W a D N A 2017 Analyzing Interaction Flow Modeling Language in Web Development Lifecycle *International Journal of Advanced Computer Science and Applications (IJACSA)* **8**

[8] Aragón G, et al. 2013 An analysis of model-driven web engineering methodologies *International Journal of Innovative Computing, Information and Control* **9**

[9] de Koch N P 2001 *Software engineering for adaptive hypermedia systems* PhD Thesis, Verlag Uni-Druck, Munich

[10] Preciado J C, et al. 2008 Designing rich internet applications combining uwe and rux-method *Web Engineering ICWE'08. Eighth International Conference on* 148-54

[11] Wakil K, et al. 2014 Enhancement of UWE navigation model: Homepage development case study *International Journal of Software Engineering & Its Applications* **8**

[12] Aguilar J A, et al. 2015 An Analysis of Techniques and Tools for Requirements Elicitation in Model-Driven Web Engineering Methods *Computational Science and Its Applications--ICCSA ed: Springer* 518-27

[13] Wakil K, Jawawi D N 2015 Model driven web engineering: A systematic mapping study *e-Informatica Software Engineering Journal* **9** 107-42

[14] Lang M 2005 A critical review of challenges in hypermedia systems development *Information Systems Development, ed: Springer* 277-88

[15] Aragón G, et al. 2013 An analysis of model-driven web engineering methodologies

[16] Lang M, Fitzgerald B 2006 *New branches, old roots: a study of methods and techniques in web/hypermedia systems design*

[17] Escalona M, et al. 2007 The treatment of navigation in web engineering *Advances in Engineering Software* **38** 267-82

[18] Consortium U 2001 *Requirements Elicitation: Model, Notation, and Tool Architecture* ed: Ubiquitous Web Applications Consortium (Deliverable D6)

[19] Ceri S, et al. 2000 Web Modeling Language (WebML): a modeling

language for designing Web sites *Computer Networks* **33** 137-57

[20] Preciado J C, et al. 2007 An Approach to Support the Web User Interfaces Evolution *AEWSE*

[21] Rossi G, et al. 2007 *Web engineering: modelling and implementing web applications* Springer Science & Business Media

[22] Stearn B 2007 XULRunner: A new approach for developing rich internet applications *IEEE Internet Computing* 67-73

[23] Luinenburg L, et al. 2008 Designing web content management systems using the method association approach P*roceedings of the 4th International Workshop on Model-Driven Web Engineering (MDWE 2008)* 106-20

[24] Basso F P, et al. *Study on Combining Model-Driven Engineering and Scrum to Produce Web Information Systems*

[25] Daniel F, et al. 2006 Combining conceptual modeling and active rules for the design of adaptive web applications *Workshop proceedings of the sixth international conference on Web engineering* 10

[26] Huang W, et al. 2010 A novel lifecycle model for Web-based application development in small and medium enterprises *International Journal of Automation and Computing* **7** 389-98

[27] Rivero J M, et al. 2014 Mockup-Driven Development: Providing agile support for Model-Driven Web Engineering I*nformation and Software Technology* **56** 670-87

[28] Lang M 2002 Hypermedia systems development: do we really need new methods *Proceedings of the Informing Science+ IT Education Conference, Cork, Ireland* 883-91

[29] UWE 2015 MagicUWE - UWE Plugin for MagicDraw. Available: http://uwe.pst.ifi.lmu.de/toolMagicUWE.html

[30] WebRatio 2015 WebRatio. Available: http://webratio.com/

[31] Escalona M J, Aragón G 2008 NDT. A model-driven approach for web requirements *Software Engineering, IEEE Transactions on* **34** 377-90

[32] IWT2I 2015 Available: http://www.iwt2.org

[33] Enterprise Architect 2011 Available: http://www.sparx.org/

[34] Koch N P d 2001 *Software engineering for adaptive hypermedia systems* PhD Thesis, Verlag Uni-Druck, Munich

[35] Koch N, Kraus A 2002 The expressive power of uml-based web engineering *Second International Workshop on Web-oriented Software Technology (IWWOST02)*

[36] OMG O 2007 *Unified Modeling Language (OMG UML)* ed: Superstructure

[37] Torres Bosch M V 2008 A web engineering approach for the development of business process-driven web applications

[38] Colin A, Kühne T 2001 *The essence of multilevel metamodeling* «UML» 2001—The Unified Modeling Language. Modeling Languages, Concepts, and Tools, 19-33

[39] Atkinson C, Kühne T 2001 The essence of multilevel metamodeling *«UML» 2001-The Unified Modeling Language. Modeling Languages, Concepts, and Tools* **2185** 19-33

[40] Koch N, Kraus A 2003 Towards a common metamodel for the development of web applications *Web Engineering* 419-22

[41] Kraus A, et al. 2007 Model-Driven Generation of Web Applications in UWE *MDWE* 261

## Authors

**Karzan Wakil, Iraq**

**Current position, grades:** lecturer at the University of Human Development-Iraq and Institute of Training and Educational Development in Sulaimaniyah-Iraq, received BSc.

**University studies:** Degree in Computer Science from Salahaddin University-Iraq-2006 and M.Sc. in Computer Science from University Technology Malaysia (UTM), Malaysia, 2013. Currently he is PhD student at UTM Malaysia His research areas are; Web Engineering, Software Engineering, Web Development, Model-Driven, Metamodel

**Dayang N.A.Jawawi, Malaysia**

**Current position, grades:** is an Associate Professor in Software Engineering Department, Faculty of Computing, Universiti Teknologi Malaysia (UTM).

**University studies:** B.Sc. in Software Engineering from Sheffield Hallam University, UK, and her M.Sc. and Ph.D. in the field of Software Engineering from Universiti Teknologi Malaysia. She has been an academic staff at Software Engineering Department for over ten years and she is currently serving as the Head of Department since October 2009. She is a member of the Software Engineering Research Group (SERG), K-Economy Research Alliance, UTM.