ANNALS OF FACULTY ENGINEERING HUNEDOARA
– INTERNATIONAL JOURNAL OF ENGINEERING
Tome IX (Year 2011). Extra Fascicule. (ISSN 1584 – 2673)

[1.]Adela BERDIE, [2.]Mihaela OSACI, [3.]Robert RAICH, [4.]Daniela CRISTEA

THE COMPONENTISATION EFFICIENCY IN REALIZING A WD ABAP PROJECT

[1-2.]POLITEHNICA UNIVERSITY OF TIMISOARA, ENGINEERING FACULTY OF HUNEDOARA, ROMANIA
[3-4.] CELLENT AG, STUTTGART, GERMANY

ABSTRACT: In this paper, we realised a study regarding the efficiency of using the componentisation technique with faceless component, in developing a Web Dynpro ABAP project. The componentisation is possible due to the architecture of the Model View Controller, the base of the Web Dynpro ABAP framework. According to the faceless componentisation technique, a Web Dynpro ABAP project is made of different components, the basic three components being: the *model* that contains the data model, the *view* component in charge with the visual part, and the *controller* component that links them and contains the testing web application. The realised case study was ordered by a company that offered the required support in selecting and recruiting the work force. The Web Dynpro ABAP application is realised with SAP NetWeaver Application Server ABAP 7.0 - trial version.
KEYWORDS: integrated system, web programming, web dynpro ABAP application

❖ WEB DYNPRO ABAP – CONCEPT AND ARCHITECTURE

The current standard for realising web applications in the ABAP environment is represented by the SAP NetWeaver concept, named *Web Dynpro*. This one is a framework integrated into the ABAP Workbench (SE80 transaction) that contains the required execution environment and the graphical development environment with special Web Dynpro tools.

The Web Dynpro architecture is based on the Web Dynpro component. The WEB DYNPRO COMPONENT is the programmable and reusable entity which represents the „heart" of a Web Dynpro application [3]. It can be considered as a unilateral representation of its parts and can be executed only through a Web Dynpro application. The parts included into a Web Dynpro component are: component Interface, component controller, component usage, window and view (Fig.1) [4].

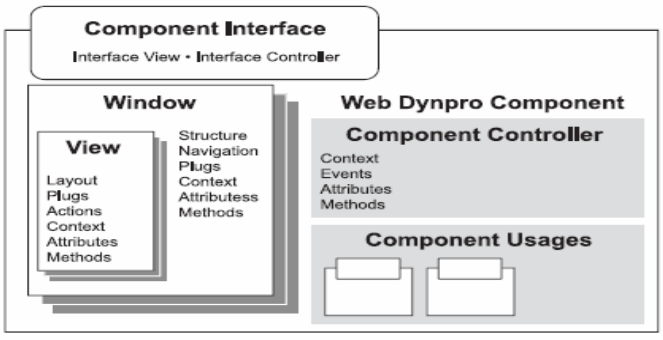

Fig.1. Parts of a Web Dynpro component [5]

The component interface enables the component to interact with other components, being made of a visual part (interface view) and a programmable part (interface controller). The component controller is in charge with the data transfer and the links among the parts of the component. By defining a component reuse, then in a component can be used other components. The View is the part where a visual interface is defined to the user. To be able to be visualised, it must be integrated in a window. The Window is the container of views among which the navigation is realised; it has an interface view uniquely associated and connected to the test application Web Dynpro.

This framework is realised according to the Model View Controller (MVC) paradigm. The MVC is an architectural model that divides the implementation of an application in three components: the model component, the view component and the controller component. The model component is in charge with data processing and returning the result to the controller; the view component is in charge only with the interface displayed to the user, and the controller component is in charge with the evaluation of the requests, sending data and instructions to the model component, and sending data to the view component. In principle, this controller component is in charge with the interaction between the view component and the model component (Fig. 2).

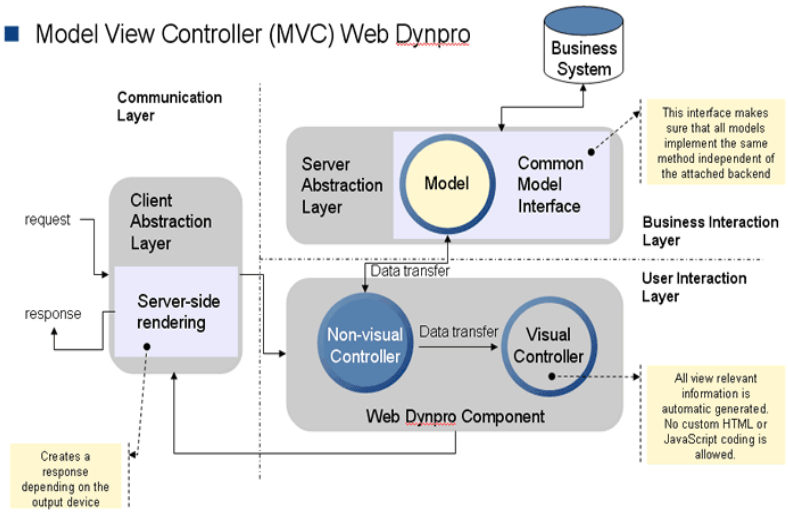

Fig.2. The architecture of the Web Dynpro framework [2]

With the Web Dynpro framework, we can develop user interfaces by using two techniques: declarative (when the interface structure is known before the execution) and dynamic (when the interface structure is partly known during the execution).

The client implementation can be defined for the web browser (Server-Side Rendering) and XML. The implementation of the metadata and the generation of HTML pages with integrated JavaScript functionalities are required for the web browser, and the implementation of XML is currently used for eCATT scenarios (extended Computer Aided Test Tool) and for client integration (*SP Smart Board)* [1].

Some of the advantages, offered by Web Dynpo in developing web applications, are [1], [3], [4]: possibility to use graphical tools, large offer of technologies, e.g. HTTP, HTML, CSS, XML and client-side scripts that are the base of each Web application, strictly separation between the data presentation and processing, possibility of using and reusing the components, easy modification of the application due to the tools it disposes of, possibility to access the data from the application context that remain intact even if the page is changing, automatic transportation of the data through data binding, automatic verification of the inputs, access to the user interface, totally integration in the ABAP development environment.

❖ THE CONCEPT OF COMPONENTISATION WITH FACELESS COMPONENT

The componentisation is the concept used to structure a large project in Web Dynpro components. The componentisation technique is based on three such components: the *View* component, in charge with the data presentation to the user, the *Model* component, in charge with the logical part of data programming, and the Controller component, in charge with the link between the other two components – *Model* and *View*, the base of the project testing application.

The *View component* will integrate all the components included in the graphical part of the project, defining a usage for each one.

The *Model component* is a Web Dynpro component without graphical elements, i.e. without *window* and *view* (faceless). This component is used to realise the logical part and to structure the data used in the entire project. We can use a faceless component [4] for a componentised application if more than one component of the project uses the same data, and if we want to extend the project.

According to the programming paradigm *Model View Controller* of Web Dynpro ABAP, it is possible to detach the data model (the logical part of the project) from the visual part. This step requires a quite big programming effort, but ensures a high transparency and quality to the project. Through this process, the componentisation methods used in the project, which need to be visible throughout the project, are moved in the *Model* component controller, as interface methods.

The great win of the componentisation based on the design pattern *Model View Controller* is that the individual components of a project are interdependent, can be easily changed and extended, can be reused, the entire project being able to be extended and easily managed and, the last but not the least, we have to mention the possibility to use in common the data models.

❖ THE COMPONENTISATION TECHNIQUE IN REALISING A WEB DYNPRO PROJECT

This technique has been implemented based on a case study realised for a company that wanted to put at the visitors' disposal information regarding the activity field, the services and the products of the company, and to offer the available working places to those people who wanted to join the company. The Web Dynpro ABAP application was realised on the *SAP NetWeaver AS ABAP 7.0* trial and consisted of Web Dynpro components.

The steps in realising a complex project, based on the faceless technology, are: realising the database, implementing and preparing the components to be linked in the componentisation structure,

preparing the componentisation structure by realising the three components: view, model and controller, linking the components in the structure and detaching the model from the visual part.For realising the database, we need to create and link the tables in the DDIC data dictionary, by using the SE11 transaction (Fig. 3).

The diagram of componentisation with faceless component, used for this project, is presented in Fig. 4.

The realisation of the project starts from the main *view* component that defines its components as usage, and realises the entire interface with the user. The data migration is made through direct external mapping. The *Model* or *faceless* component includes the logical part of the project: the common nodes of the components that compose the visual part, the methods of populating these nodes, the interface methods and other methods adequate for the *view* components [6].

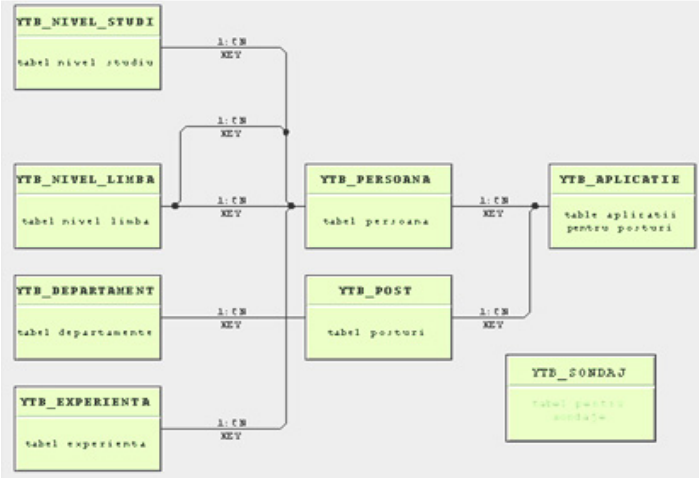

Fig. 3. Database table links

Because the navigation from one page to another is the basic element of any web application, the windows of the components embedded into the main *View* component will define outputs (outbound plugs) and inputs (inbound plugs) used to realise the navigation diagram . At the *view* level, the outputs will be linked with the inputs defined in the *window*. The link between the window inputs and outputs is logically realised, by using the interface event handler methods that correspond to the defined inputs.

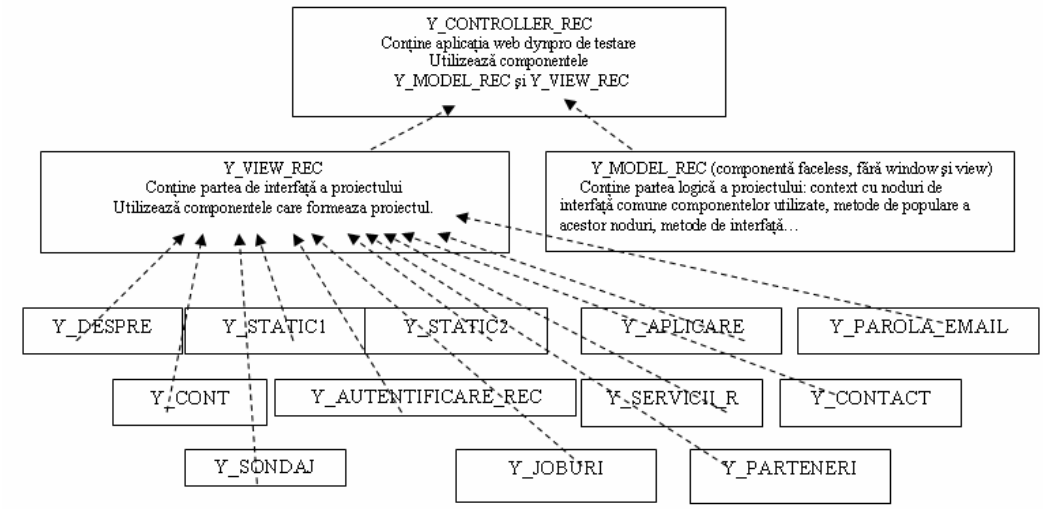

Fig.4. Diagram of componentisation with faceless component

The *Controller* component links the components *Model* and *View*, defining them as usage, to ensure the data traffic.

Regarding the separation of the data model from the visual part, in the *Model* component, besides the methods used to populate the interface nodes that correspond to the reused components, there are brought all the methods along with their afferent codes from the component controller of the used components which, even without navigation sequences, want to be visible in other components. The reused components will keep the name of the respective methods, but these methods will trigger only interface events. The controller component that links the components *View* and *Model* includes the event handler methods able to call these methods.

❖ CONCLUSIONS

Web Dynpo is one of the top technologies used to realise high quality web applications, and the componentisation is the technique used to build web projects on the nowadays SAP platform. The faceless componentisation, as a technique based on the design of the *Model View Controller* pattern, enables the independence of the components along with their interchange and reuse and, in the same time, the easily extension of the project. Although this requires a greater programming effort, it offers a maximum win.

Comparing with other frameworks (e.g. Prado), where the physical creation of the file that runs the application and the interface file are required, the Web Dynpro applications need only the realisation of a component whose window will be attached to the user. The application structure is

realised without writing a single line of code, and the interface with the user is a view built with view design elements and programmed according to the requirements.

In conclusion, the Web Dynpro framework is not a tool for creating presentation pages with much animation and sophisticated graphical elements, but for implementing and realising latest generation business applications.

❖ REFERENCES

[1.] Ulli Hoffmann, *Web Dynpro for ABAP*, Publishing House SAP Galileo Press, Bonn, Germania, 2007.
[2.] SAP Developer Network (http://sdn.sap.com).
[3.] SAP Portal - Help (http://help.sap.com).
[4.] Web Dynpro for ABAP (WDA, WD4A, WDF4A), Release NW2004s SP8 (SAP AG Online Help 2006), available in electronic format (pdf) on Internet network.
[5.] Ulli Hoffman, *Get started developing Web – native custom SAP application with Web Dynpro ABAP*, SAP Professional Jurnal, 2007.
[6.] Web Dynpro ABAP Programming Guidelines (SAP AG 2006), available in electronic format (pdf) on Internet network.

ANNALS OF FACULTY ENGINEERING HUNEDOARA
– INTERNATIONAL JOURNAL OF ENGINEERING