Stop Generating your User Interface! Start designing IT!

Lonneke Dikmans
Vennster
The Netherlands

Keywords:

User Experience, UX, Personas, Scenarios, Interaction Design, Usability test, Business Process Management, BPM, BPMN, Oracle BPM Studio, BPEL

Introduction

Process analysts and developers often make the mistake of equating user interaction with process flow. This is even more abundant now that we can execute the BPMN 2.0 models that we create. Analysts tend to model the user interaction in the process and then ask the developers to generate/build a screen for every human task that is modeled. This results in user interfaces that are hard to use, error prone and inflexible and in processes that are very hard to change. Saving money in the project by generating a screen for every human task flow will cost the organization a lot of money for several reasons that will be explained in the session. A better approach is to design the process flow and to design the user interface(s) separate from each other. This way you will be more flexible both to make process changes and to make interface changes.

Case: Building permits

An application that was built to support a new permit process was created. The following roles were identified:

- Applicant: a company or person who wants to build something;
- Front office that receives the application and communicates with the applicant;
- Building inspector who reviews the application;
- Finance department who sends the invoice;
- External advisors that the building inspector can consult. For example, the fire department.

The software was realized using Oracle SOA Suite10g, based on a process design in BPMN 1.1. The business logic consisted of a combination of automated and human tasks. This was realized using BPEL, Web Services, the Task Service and Java Server Faces (JSF) for the front end. The process was rather fine grained and the user interface was generated based on the XML of the Human Task that was defined.

The problem

This way of creating an application is very fast and therefore looks like a good way of saving cost. However the organization experienced two problems with this approach, one from a process perspective and one from a user experience perspective.

Process perspective:

1. The process was very fine grained, because every step in the user interface needed to be a step in the process. This meant that a change in the user interface also needed a change in the process.

2. The BPEL process carried a lot of data, because all the data of the task came form the BPEL process. This means that the business process was cluttered with assign statements, copying values from one activity to the next. These attributes did not have any impact on the process flow.

User experience perspective:

1. The software is organized completely from the 'process unit' perspective: the application for a building permit. This is fine for the applicant and the front office, but building inspectors like to organize inspections based on the address so that they can efficiently visit sites; The same is true for the decision makers: they like to group applications by date, so they can have the meetings (if necessary) in time.
2. Users have little flexibility and access to a limited set of data: just the data that is carried by the process.

In this case the software was much more expensive than budgeted once it was taken into production. There were two problems:

1. High administration cost. Because the process was so fine grained a lot of effort needed to be spent to keep the system up and running. If users made a mistake and they wanted to change data, they needed a technical administrator to change something in the process flow, or to restart a process.
2. High cost to change something in the application. Whenever a change was needed, all layers of the application needed to be changed, because the UI and the process were tightly coupled. Most change requests were related to user interface improvements. Impact of these changes was very high because the data in the UI were based on the data sent by the BPEL process. A new field in the UI meant changing the process.

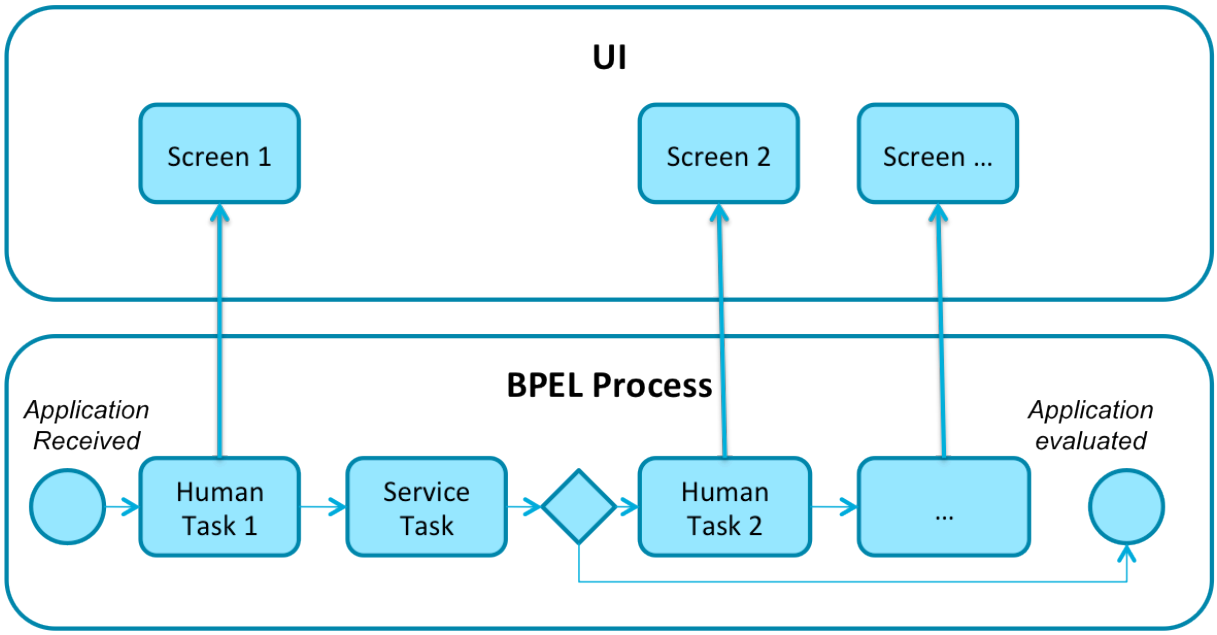

Illustration 1. Tightly coupled UI

Case: employee self service and manager self service (ess and mss)

An organization wanted to save money by offering employees and managers self-service for HR related data. In the current situation, people from the payroll department would enter expense data, change bank account numbers etc. The self-service application would make it

possible for people to enter those data themselves, saving a lot of cost in administration. One of the most important requirements was that it would be easy for the people to enter the data. The user population was heterogeneous: from blue-collar workers to local government officials. A pilot to compare three solutions was defined for the expense process.
The following roles were identified:

- Employee who want to submit expenses;
- Managers who need to approve the expenses;
- Pay roll administrators who check the expense against tax rules and pay the expenses.

The process consisted of a combination of manual and automated steps. The organization used E-Business Suite 12 for HR processes. The data about the employees and managers came from EBS. The end result (expense) should be stored in EBS as a pay element. The pay roll department would process it from there.
There were three possible solutions:

- Use the self service module from EBS;
- Buy an off the shelf product offering integration with EBS;
- Build something specific for the organization.

There were several criteria important for the decision, but the most important ones were user friendliness and cost. Therefor the approach incorporated UX from the start, in the requirements gathering process.

The project executed the following steps:

1. Define and create the business process based on BPM techniques
2. Design and create the user interface techniques
3. Integrate the two based on architecture techniques

Note that the steps were partly executed in parallel.

The result

The process was designed based on BPMN method and Style (Bruce Silver). It started with determining the ‘unit of execution’. There was some discussion whether this would be an expense (trip), or expense lines (travel, meal), or all expenses that occurred in a month. Here the separation between UX and process really helped. Because the UI would not be based on the process directly, we could analyze the process and determine that the unit of control is an expense line. The manager could approve the travel, but decline the meal and vice versa, without the employee having to reenter all the data from the trip.

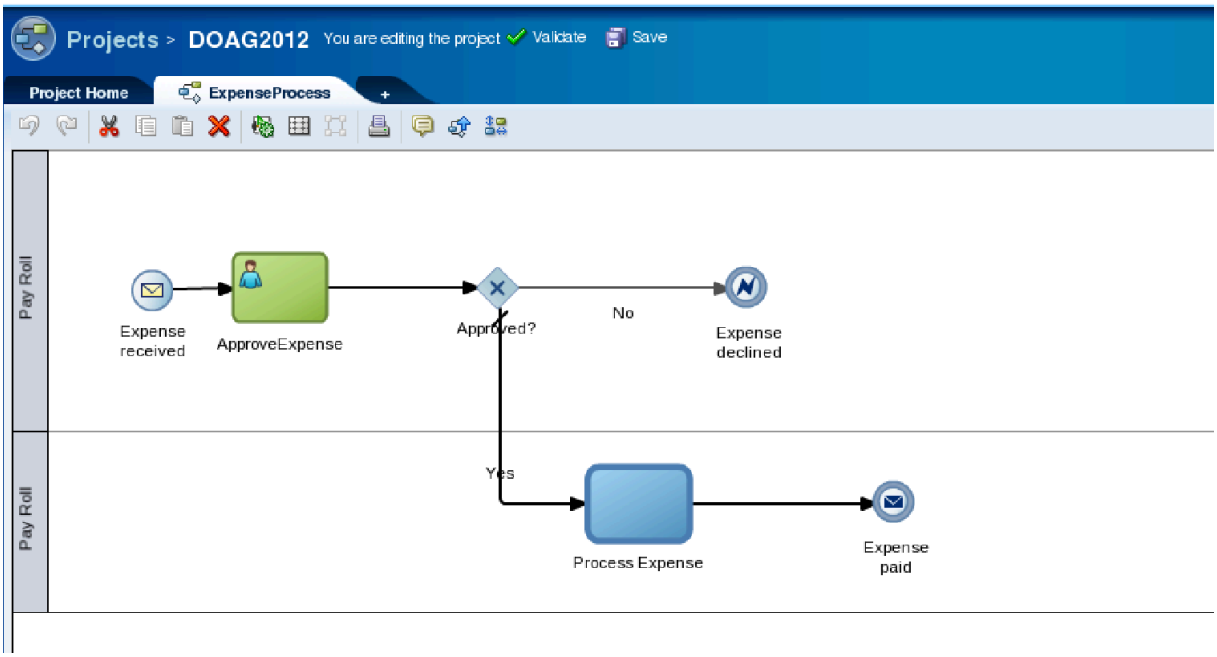

Illustration 2 Expense Process

The user experience was designed, developed and tested using familiar user experience techniques:

- Personas and scenarios based on the roles and the different types of users in the organization;
- Interaction design and prototypes;
- Usability testing to improve the design.

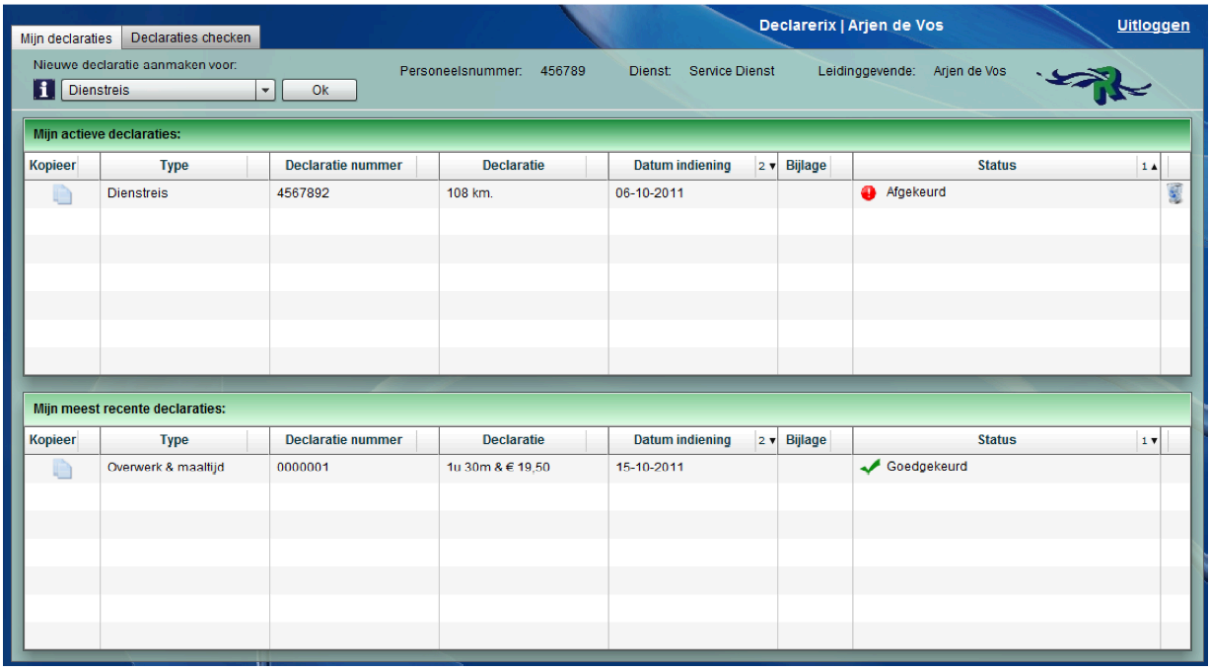

Illustration 3 Screen shot of the prototype

The different parts were integrated based on services. The process was connected to Oracle E-Business Suite using PL/SQL web services. The services, the process and the user interface were connected using REST services.

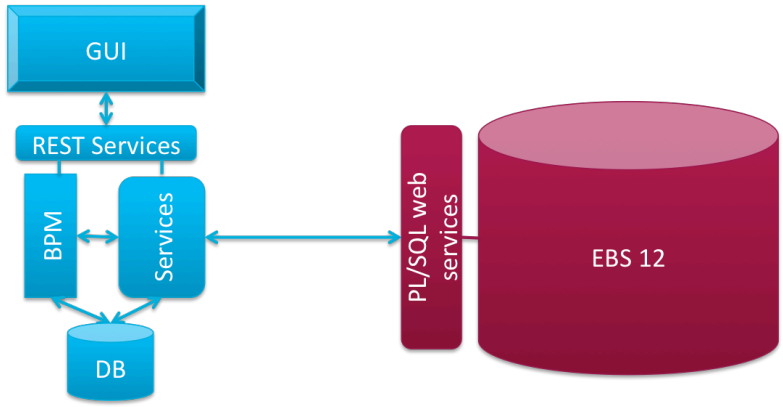

Illustration 4 Architecture overview

Conclusion

The chosen approach proved to have several advantages:

- The Application appeals to users and satisfies there needs;
- The Process is monitored and executed in compliance with tax rules;
- Parallel execution of different disciplines because of the loosely coupled architecture, so no delay in the project;
- Extra cost in beginning of project, but less cost in maintenance because it is easy to change and there is less need to change.

Contact address:

Lonneke Dikmans
Vennster
Agnes van Leeuwenberchstraat 15
3515AX Utrecht

Phone: +31(0)6-15083349
Email lonneke.dikmans@vennster.nl
Blog http://blog.vennster.nl
Internet: http://www.vennster.nl