# Designing dynamic processes

**In this design guide**

## Introduction

Today's business processes have to address many, complex requirements. Governments and enterprises offer personalized and contextual products, whereas customers and citizens all have their own unique demands. As a result, the business processes for selling and offering these products are divers and contextual as well. At the same time, regulations in the area of compliance and a growing rate of change introduce additional complexity. These developments pose major challenges to the field of business process modeling (BPM). Conventional process modeling, in terms of activities and the flow they are executed in, has proven to lead to complex and often rigid business processes. Dynamic processes, as part of dynamic case management, are an answer to the problems and challenges that conventional business processes face.

## Dynamic processes

Dynamic processes are processes in which not all possible scenarios are completely delineated upfront. In dynamic processes there is no diagram, model or code available that tells us exhaustively in which order possible scenarios can occur and what to do when in each scenario. Instead of meticulously connecting activities to establish completeness (or define ordered sequence) in flow, in dynamic processes all activities can in theory contribute to the process at any given time. For each activity that is part of the process, a precondition simply states when the activity can, should or must  be performed. Such a precondition is a boolean statement that makes use of the available data, the artifacts in the system and the state or phase the process is in. A dynamic process engine (like Blueriq) will decide at runtime which activities can and must be performed at any given moment in the process. In Blueriq, dynamic preconditioned activities are modeled as ad-hoc tasks.

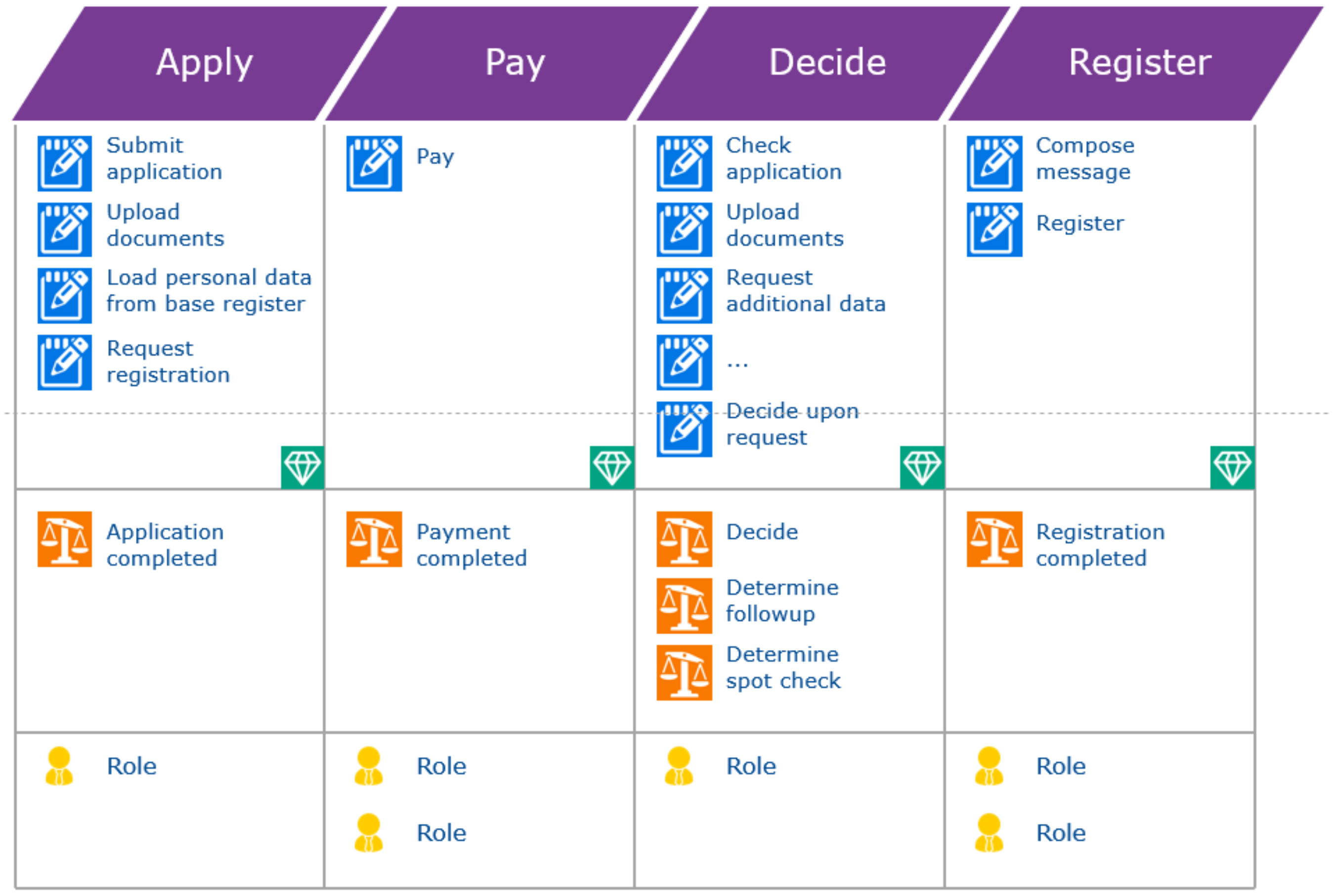
From a Blueriq point-of-view, it is not mandatory to use preconditions and ad-hoc tasks exclusively, nor is it mandatory to use flow exclusively. One of the unique aspects of Blueriq is that both approaches are available and can be intertwined to get the best of both worlds. So whenever there is an activity that may only be performed when another activity is completed, it can be modeled with flow. Many discussions in the field are about the difference between static and dynamic processes. By allowing both, we allow for a hybrid approach.

## Design

Some people are concerned that - when using ad-hoc tasks - the business process model becomes more difficult to understand and maintain than when flow is used. In situations where no good design is made upfront, they might very well be right! If a process with a lot of activities is modeled by simply dumping all these activities in one single batch and just create (complex) preconditions for all of these activities, this will have, at model time at least, no benefit over a flowchart.

However, in business processes it is very uncommon that all activities can be applicable throughout the entire process. At a certain point (in dynamic case management terms: when a certain milestone is reached), some activities are simply not part of the possible process anymore. Or not part of the process yet. This is why business processes can be dissected into phases. Using phases, milestones, ad-hoc tasks and preconditions will result in really dynamic and flexible business processes, without the downside of delineating all scenarios upfront and connecting the activities with all possible paths.

Below a schematic outline of such a process with phases, activities and milestones is shown.



It is not really necessary to understand every tiny detail of the process diagram shown above. It is more important to understand the different dynamic case management *concepts* behind it:

- Phase
- Task/Activity
- Milestone
- Precondition

These concepts - and depending concepts - are discussed below.

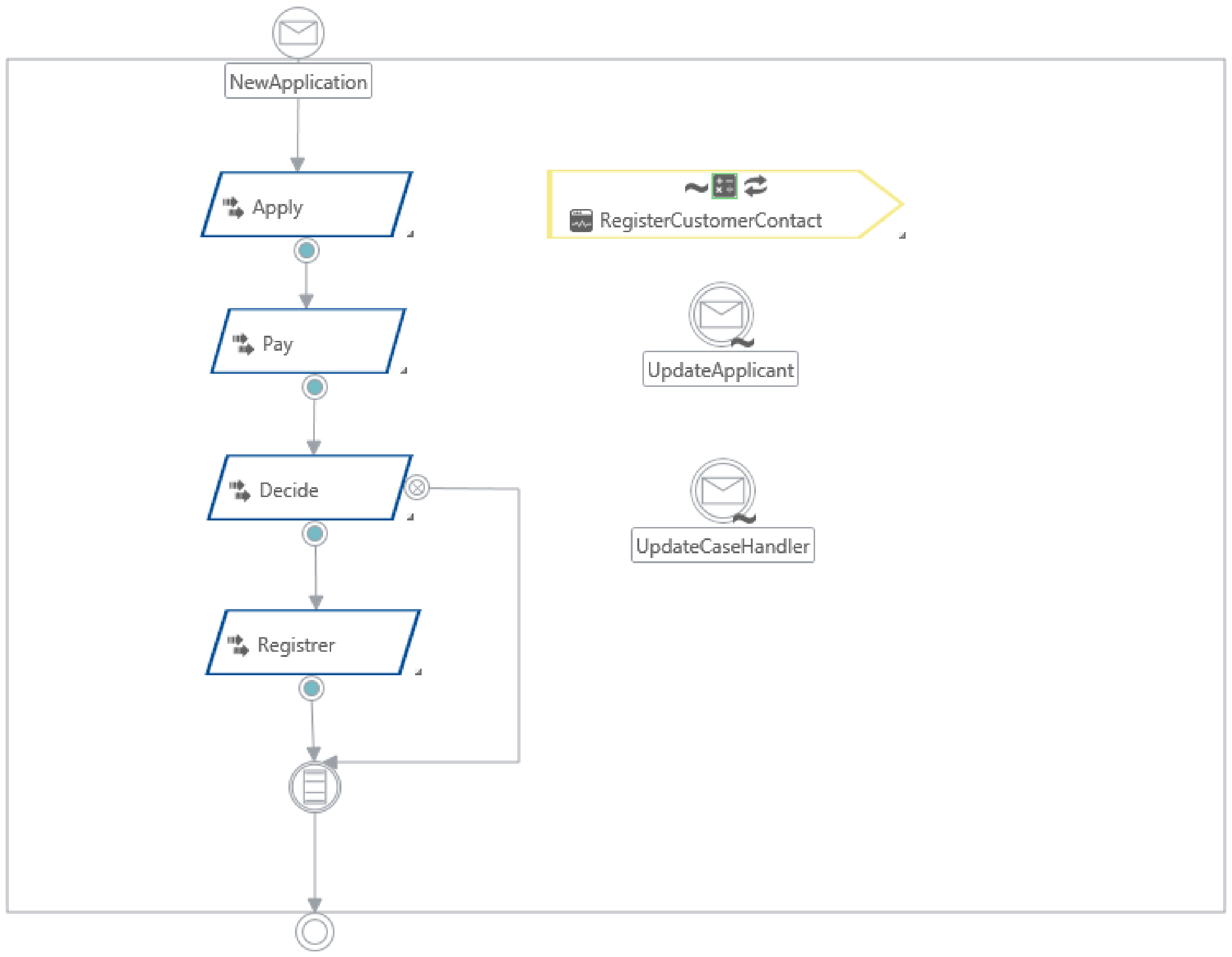
An important aspect of the process design shown here, is that at the highest level, we speak of phases rather than processes. A phase resembles a period of time within the entire process when the end user is free to perform tasks that are applicable in that phase and contribute to the goal of the phase. Tasks that are not applicable anymore or not yet are not shown, making the application more understandable for the end user. The goal of the phase is called a milestone (in this diagram depicted by a diamond 💎). Whether a milestone is reached or not is merely a decision, based on availability of data and artifacts in the system.

The tasks within a phase are mostly ad-hoc or automated, but this is not mandatory. Blueriq has the ability to mix ad-hoc and automated tasks with flow-oriented tasks. The most important aspect of this process design is that , as much as possible, the availability and necessity of the tasks depends on the availability of data and artifacts, and not on the completion of other tasks.

As mentioned before, phases consist of tasks. These tasks produce data, which is entered into the system, for instance about an applicant or about a desired product. Tasks also produce artifacts, such as uploaded documents, pictures etc. and generated documents. Preconditions are decisions that tell when the tasks can be (or even must be) performed. These decisions are all based on data, artifacts and other decisions.

In the design shown above, also a row is reserved for the roles that participate in each phase. Although this is absolutely necessary in business process modeling, it is left out of scope in this article.

The Blueriq-design of the example of the process with four phases shown above, is given below.



The process shows different dynamic case management concepts, such as phases and different types of tasks.

## Phases

Most processes are - from a high point of view - very linear. So is our example: it consists of the sequential phases Apply, Pay, Decide, Register. In the vast majority of business processes, the phases are singular and linear, meaning a case or process will always be in one phase and after that phase, another phase will follow. When during the design of phases it turns out that many phases can be applicable at once and many different phases might follow a phase, the granularity of the phase is probably off: you are most likely designing tasks/activities instead of phases.

Sometimes however, a phase can be skipped. In this particular example, when registration is rejected on grounds of inadmissibility, the phase registering is skipped altogether. The rejection because of inadmissibility is done when the application is not complete or documents are not valid and the applicant has not been able to provide the necessary information. When the application is complete and all documents are valid, registration could still be denied. Denying and rejecting sound alike, but differ in this example. Denying an application is done when the application is not eligible, although complete and valid. Non-eligible applications are registered as well as eligible applications, but with a negative decision. Rejections, as mentioned before, are not registered at all.

## Generic tasks

Sometimes there are tasks that are always available. In this example, it is always possible to register some sort of contact between applicant and the back office, for instance by mail or telephone or in person. This is done using an ad-hoc (~) task that is also a repeatable task (⇄) and has no precondition (▦) throughout the process. In Blueriq, an ad-hoc task without a precondition is modeled with a precondition that is always true.

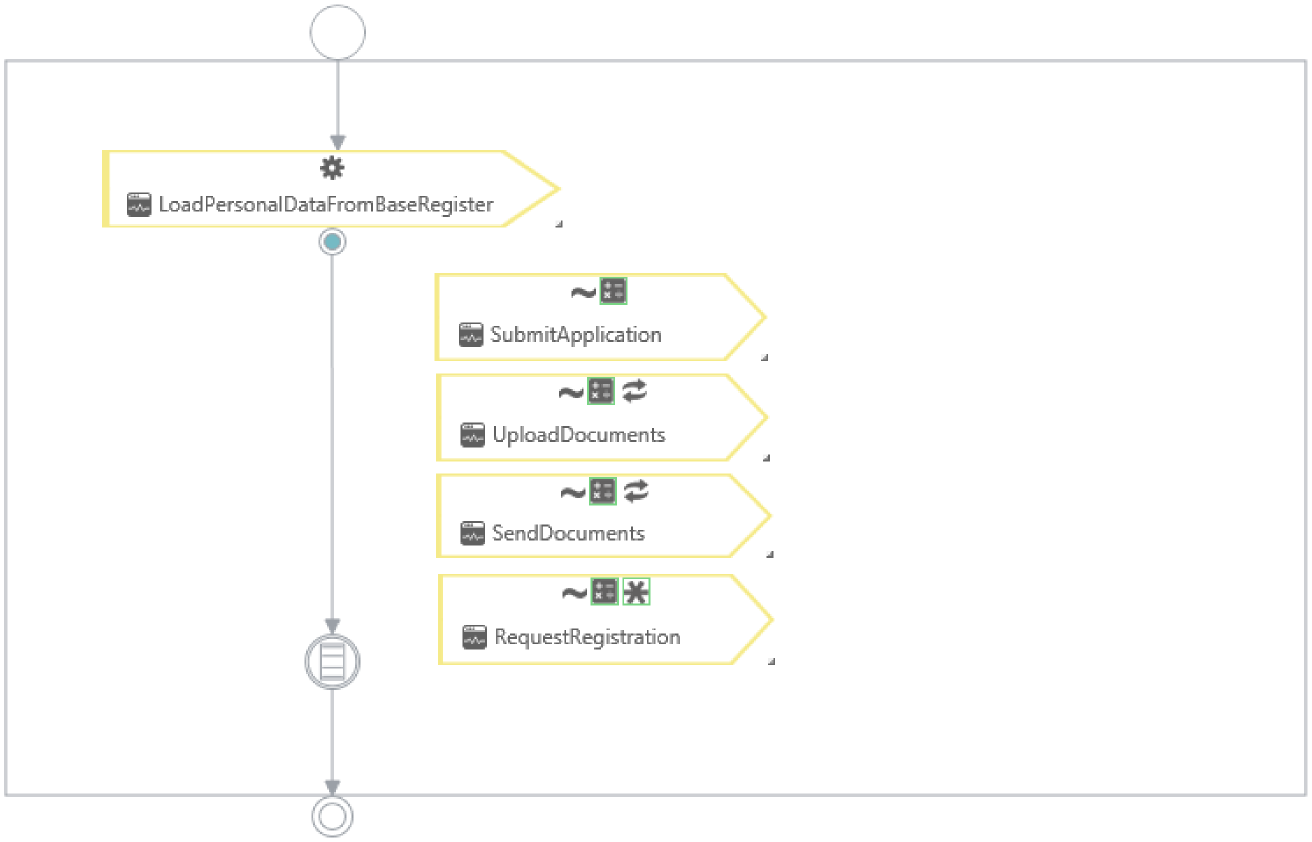
## Life events

Events such as relocation or death of the applicant or, in other cases divorce, the death of a partner, children moving out or unemployment, will trigger the process from outside. For these life events, message events are used.

## Process end

After the final phase, this particular process may not end. In this design, a conditional node (that will never ever become true) has been added to keep the process from actually ending.
(In Blueriq, ending a process will end the Blueriq-application).

Shown below is one particular phase of our example process (Apply), containing five tasks.



The process above shows another set of dynamic case management concepts.

## Hybrid process: Automated, Manual and Ad-hoc tasks

In the process diagram of the first phase (Apply), we see a hybrid process, containing automated tasks (⚙) as well as manual tasks. Furthermore, we see flow-oriented orchestration of these tasks by means of arrows and ad-hoc orchestration by means of preconditions. This is a very strong feature of this process design; with Blueriq it is possible to design processes that contain the best of both worlds. The automated task LoadPersonalDataFromBaseRegister must always be performed first, this is why the task is connected to the start event of this phase by means of a flow-arrow.

## STP support

Because of the fact that only automated tasks are part of the flow of the Apply phase, straight through processing (STP) is fully supported. Beware that when non-automated tasks are placed within a flow, STP support is no longer possible. The choice to disable STP in a phase has large consequences and should be made by weighing all aspects of the design. Many customers want STP support and manual support within the same process, which means that only automated tasks are possible in a flow.

## Repeatable tasks

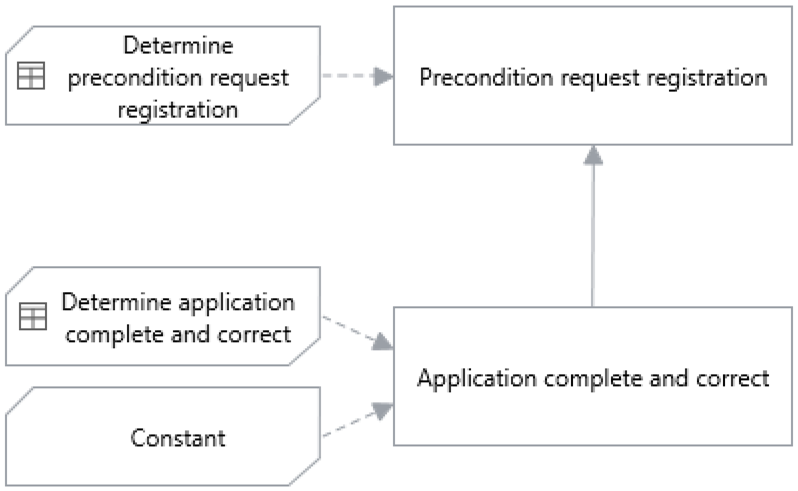
Some of the tasks of the phase Apply are repeatable (⇄), such as UploadDocuments. It is always possible to upload documents within the Apply-phase, even after documents have already been uploaded. There is no precondition on this task, so it can always be performed.

## Mandatory tasks

In this phase, there is one mandatory (✱) task, RequestRegistration. A precondition will determine whether the task may be performed or not, but the phase Apply will never end unless the task is performed.

## Preconditions

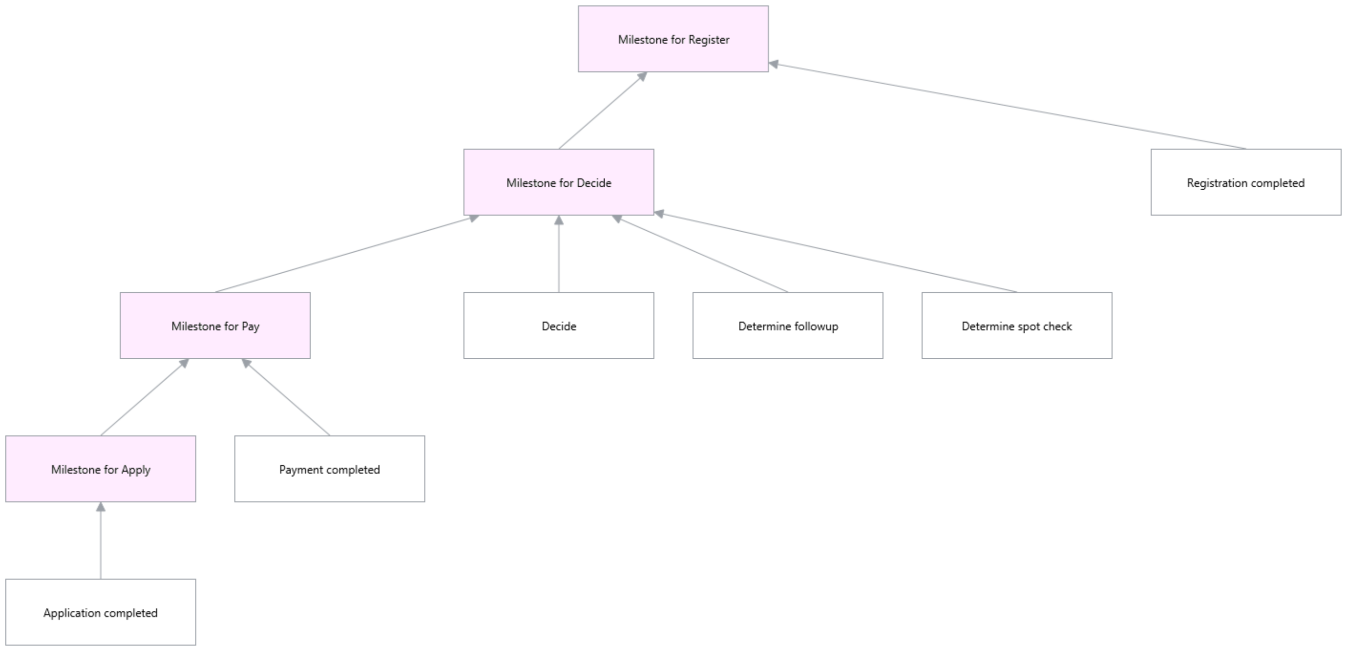
All tasks can be performed in a phase, but the order in which they may or must be performed is not set in stone, neither is the applicability of these tasks. Preconditions tell us when the tasks may be performed. A Decision Requirements Graph can be used to visualize the precondition, as shown below for the task RequestRegistration.


Determine precondition request registration
Precondition request registration
Determine application complete and correct
Constant
Application complete and correct

**Milestones**

Milestones (in the process diagram depicted by a diamond ) are used as goal for each phase. A milestone is merely a decision and is based on data and artifacts. The milestone for the phase Apply for instance is only true (and therefor reached) when the application is complete, meaning all mandatory data is provided and necessary documents are uploaded. Milestones are used to steer the phases at a high level. In theory each phase has one milestone that ends the phase and each phase (except the first one) will only be opened when the milestone of the previous phase is reached. It is very uncommon and not advisable to design a process that can be in two phases at once or no phase at all.

Milestones are merely decisions. As mentioned in a previous article (How to use Decision Requirements Graphs to visualize ad-hoc tasks in business process modeling), a decision requirements graph can be used to visualize the entire process. See below and compare this diagram with the first diagram of this article.



## Benefits of the design

The main question one might ask is why the design described here should be used. What are the benefits?

**Flexibility**

First of all, this design is really flexible, similar to the real world the end users live in. When an activity is added to the process, all that needs to be done is add the task to the proper phase and determine its precondition. That's it. In conventional business modeling with activities and flow, adding an activity is much more complex; most likely the activity can or must be added in more than one part of the flow. This re sults in large and expensive migration to get from one process version to the next.

#### Understandability

Secondly, this design is much easier to read for business users. Flowcharts tend to be seen as 'technical' or 'IT-like', although that was never the intention of BPM. A process diagram containing phases with activities is much easier to read by business users. Instead of being responsible and accountable for dozens or even hundreds of process diagrams containing complex flow, business users are delighted to see their processes narrowed down to a few simple designs, where the complexity is not in the process itself, but in the activities.

#### Insight?

Of course there are some concerns regarding this approach. The main concern regards insight. As mentioned before, in a previous article (How to use Decision Requirements Graphs to visualize ad-hoc tasks in business process modeling), we discussed how Decision Requirements Graphs help visualize ad-hoc tasks. This article was written as a response to often heard comments on ad-hoc task modeling regarding insight and overview when modeling business processes with ad-hoc tasks rather than (work)flow oriented tasks. Using the strategy described in this article, combined with the design as outlined here, will result in dynamic processes which are not only really flexible, but also maintainable.

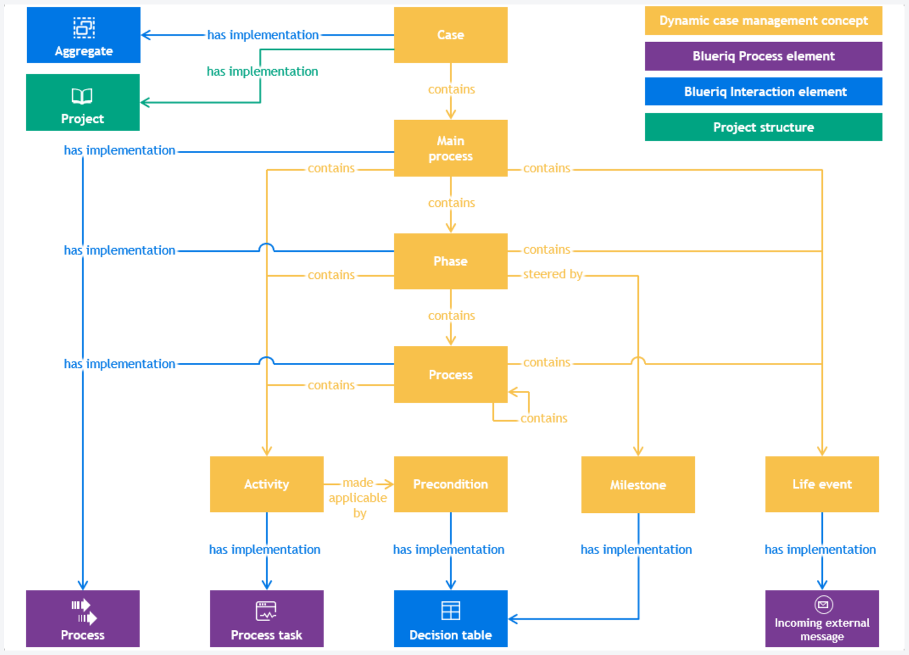
#### Execution

One should however realize that the real benefit of this approach is not only on the design of the process. In execution, flexibility is needed more and more, and knowledge workers do not do things in a fixed order, do not handle each case in the same way, and might need more freedom to deviate from the path or work with others. This is perhaps best compared to the navigation paradigm. Nobody ever prints a complete route description before stepping into his car. Even more, printing all possible routes to take is practically impossible. Instead, a modern navigation device is continuously adapting to changes; road blocks, traffic jams or other exceptions, and, just in time, calculates a new route. So why try to think of all possible routes in process design?

## See also

Everything about persistency management and aggregates can be found here: Persistency Management guide.
A visualization of most concepts discussed in this article can be found here: Blueriq visuals.
Specifically for dynamic case management, see Dynamic Case Management concepts.
A template for dynamic case management can be found here: DCM Foundation - v4 [editor] .

**Dynamic case management**



Everything about persistency management and aggregates can be found here: Persistency Management guide.
A visualization of most concepts discussed in this article can be found here: Blueriq visuals.