IT Finanzmagazin
Das Fachmagazin für IT und Organisation bei Banken, Sparkassen und Versicherungen

Strategy

Using standard software like a customised solution? How to open up banking software for workflows

Digitisation reveals a conflict that is simmering in most IT departments: is it better to use one software for all, to develop in-house or to rely on a framework? Those who are against standard software are mainly those who fear that they will no longer be able to distinguish themselves sufficiently from their competitors. Anyone offering such software should therefore open it up so that customised processes can be implemented. A practical example from the banking world.

At first glance, the dilemma can hardly be resolved. On the one hand, the companies want to set themselves apart from each other in the digital competition. They advertise by offering their customers the best experience. "Banking without nonsense," promises a well-known smartphone bank from Berlin, which has rapidly gained new customers in recent years. On the other hand, however, this also means developing a great deal of the software used in-house, so that the offer also feels different from that of other banks. Some say that financial institutions must even transform themselves into software companies. Major banks such as ING or Goldman Sachs have long since seen themselves as tech companies with a banking licence. But doing everything in-house does not always pay off in every situation.

More power to the users

In areas where the same processes always happen, self-developed software is not worthwhile.This applies, for example, to accounting and finance, and often also to human resources. Because customers hardly notice any of these processes anyway, individually produced software would be too expensive. Frameworks offer a supposedly safe way out of this situation, because they provide ready-made modules for recurring tasks and also allow customised individual services to suit the needs of the customer. However, this requires many manual steps. This results in the risk of building an IT solution that is barely fit for release and creates new problems with every update of the framework.

The ideal solution would therefore be standard software that allows customers to define their own processes. The idea: a software with an integrated workflow engine (WFE) that can be controlled via a script language (see Fig. 1). The script language distinguishes between status and quality queries. Each status specifies what should happen next, and each quality query checks whether or not predefined conditions exist. This determines whether and how an individual status changes. Once the workflow is

fully described, the engine compiles the code. Within the software, the workflow engine in turn creates tasks that are then processed by a task engine. Changes formulated via the script language can thus be imported without a release upgrade.

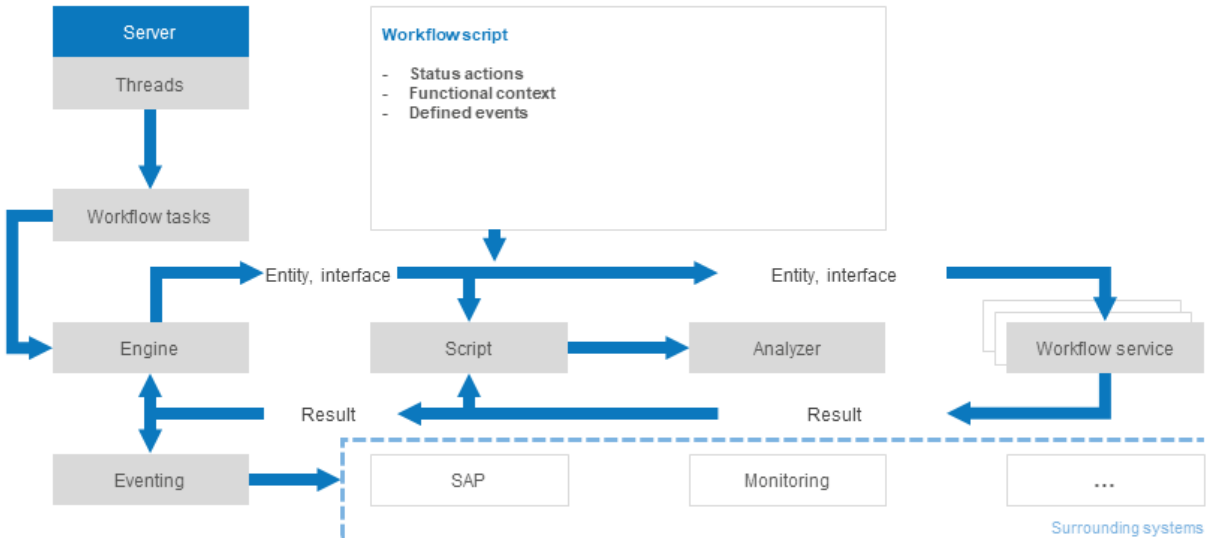

Fig. 1: Workflow script engine integrated in a standard software.

Internally, the software works exclusively with tasks; externally, it opens up via the workflow engine and the script language supplied. A large part of software development thus moves from the sphere of the manufacturer into the sphere of the company introducing the software. This can be done either directly by the in-house IT and business departments or by a company that is familiar with the respective subject matter. The technical aspect is practically separated from the functional aspect. Unlike with many frameworks, the later users do not need to make any changes to the source code itself. This eliminates one of the biggest dangers that cause many such projects to fail.

Individuality despite uniformity

This architecture is particularly well-suited for business transactions that can be broken down into individual subtasks and processed in a user-defined sequence. In financial institutions, payments are a typical use case. If money is to be transferred, the institutions must check, for example, whether the account of the ordering party is covered and whether the IBAN of the recipient is correct. But that is by no means all. Financial institutions must also check whether there is a suspicion of money laundering – especially when payments are made abroad – and whether the recipient is on an embargo or sanctions list. Although the tasks hardly differ from bank to bank, the sequence, conditions and the interfaces to other IT systems used are nevertheless very different.

The following listing shows how the payments software developed by PPI is configured to behave in the event of an incoming payment, with regard to any charges that may be incurred.

```
Status CHECKINBOUNDCHRGS {
    if payment is inbound and (payment is ourCharges) then {
        if payment is ourChargesReceived then {
            just set status VALIDATERECEIVCHGS and leave step
        }
        if payment is correspondentChargeExpected then {
            if payment is debitAuthorized then {
                just set status CREATEADVICEOFCHGS and leave step
            }
            just set status CREATECHARGEREQ and leave step
        }
    }
    just set status DONE and leave step
}
```

Fig. 2: Who pays the charges incurred by a credit transfer?

In the case illustrated above, the system accepts a payment from another bank and checks whether the charges are paid by the bank triggering the payment – in technical jargon, this rule is called OUR charges. If this is the case, the system should check whether these charges are included in the transfer in addition to the amount transferred. If so, the system creates a new status so that the charge amount is checked and retained. If not, the system asks whether the charges can be collected directly from the bank from which the payment comes. This determines which status is generated and whether the system collects the charges directly or whether a payment request is created. Depending on the client, almost any number of rules can be mapped.

This is particularly important for individual payments. Large corporations usually look for a bank in every region of the world to process all payments centrally. To do so, they set certain requirements which the bank must be able to meet in order to win the bid. As a result, the bank in question – or the software used – suddenly has to tailor an allegedly standardised payments product into a customised offer that requires attention to many details. The corporations often prescribe which banks the money is to be channelled through and at what times of the day it is to be booked. Unlike SEPA payments in the euro area, for example, cross-border payments are not as simple as mass transactions.

All or nothing?

In light of this, many banks have to decide whether or not they want to see payments as a strategic business area at all. Some banks hand over the complex business to a larger institution or outsource their payments completely, thus avoiding the problem of having to make constant statutory adjustments on a platform they may have developed themselves. Innovation is then neglected, which is a particular problem for smaller institutions, as a study by the European Banking Association (EBA) shows.

A software that can be flexibly adapted and whose manufacturer nevertheless relieves the user of all tiresome tasks is therefore particularly worthwhile. This may not only apply to financial institutions

that transfer funds from one country to another, but also to other sectors in which the legislator imposes strict regulations, such as energy, logistics or foreign trade. Companies involved in the international movement of goods must, for example, use an interface to access customs filter lists that provide information on prohibited goods or illegitimate senders. If something goes wrong in the process, they risk losing their status as "reliable shippers", which entails considerable disadvantages. In addition, they have IT systems to calculate prices, maintain their own black lists and comply with internal compliance regulations, similar to those of a financial institution.

Conclusion

In the digital age, whenever a company wants to differentiate itself from the competition in a particular business area, the focus is almost automatically on the software it uses – and on whether it was developed in-house or purchased. Purchased software is suspected of making the desired differentiation impossible, because the development often only pays off once many customers execute as many similar business transactions as possible on one and the same platform. The functionalities are therefore also standardised. Whoever succeeds in separating the functional from the technical aspects solves this basic problem. Customers can then differentiate themselves sufficiently despite using standard software and no longer need to deal with further technical developments, as those will be included in the next release.

Authors

Thomas Riedel has been a senior product manager at PPI since 2015 and is responsible for software used for core payments processing as well as clearing and settlement. Prior to his employment with PPI, the engineering mathematician worked for almost 15 years at an international consulting and IT company and developed products for payments, most recently as Head of Payments.

Mathias Seeliger is a leading software architect at PPI and is responsible for the architecture of the payments platform. The trained IT specialist has more than 15 years of experience in the development of payments software. His focus is on continuous delivery, testing with Docker and Kubernetes as well as APIs.

Courtesy of IT Finanzmagazin:
https://www.it-finanzmagazin.de/standardsoftware-software-workflows-106406/