COLUMN

Lifecycle Management Matters

SAPinsider

This article appeared in the Oct ▪ Nov ▪ Dec 2011 issue of *SAPinsider* (http://sapinsider.wispubs.com) and appears here with permission from the publisher, WIS Publishing.

WIS PUBLISHING

How to **Get the Most Out of ABAP and Java** in the Context of SAP Technology

Choosing a Java-Based Solution

by Dr. Thomas Weiss and Martin Jaekle, SAP AG

Dr. Thomas Weiss (thomas.weiss@sap.com) has a Ph.D. in analytic philosophy and joined the SAP NetWeaver Product Management Training team in 2001, where his responsibilities included the e-learning strategy for ABAP. He is now a member of the SAP TIP Core Product Management team.

Martin Jaekle (martin.jaekle@sap.com) joined SAP in 2008 to contribute to strategic decisions focusing on SAP NetWeaver Java and closely related topics. Prior to joining SAP, he was responsible for various aspects of software development and product management at different companies.

SAP leverages the strengths of both the ABAP and Java programming languages in its technology platform, and customers can use either language for their own development projects and achieve ROI — but which one *should* you choose?

The answer isn't a question of which language is better, or whether you should always choose one over the other — the right solution for you depends on the unique characteristics of your particular development project, and which particular ABAP or Java strengths best suit your project. In our previous *SAPinsider* article,[1] we covered some of the considerations that might lead a company to choose ABAP to develop a particular add-on scenario. In this second article in the series, we'll walk through two example scenarios that result in selecting a Java-driven solution.

While each example scenario leads to a particular technology choice, your choice may be entirely different, depending on your project's criteria and requirements — it may even be a mixed solution. Regardless, by using these considerations as a guide, you will be able to make a well-informed, well-supported decision, and start down the right path with confidence.

Example Scenario: Adapting Processes to Meet New Requirements

Change is endemic to business, and modifying your processes is a given part of keeping up with your customers and competitors. In addition, as markets have become increasingly dynamic, the time to develop new products, services, and processes has dramatically shortened.

With these factors in mind, suppose an insurance company wants to become more flexible in delivering new products, such as unit-linked policy sets. While these new products resemble the company's normal offerings, changes still must be made to the relevant existing processes, which are implemented in SAP for Insurance solutions and use a large amount of data from the SAP ERP system. A direct implementation in the existing system would require large adaptations of the ABAP-based SAP ERP and SAP for Insurance applications, and implementing them by changing and enhancing the ABAP code of the relevant SAP development objects would mean a very tight coupling of the add-on for the new product and the underlying SAP solution. As the new products and the related processes are likely to change, and possibly at a high frequency, a close coupling of the new add-on with the more stable SAP solution is not advisable.

So what's the best way to implement the necessary process changes in a scenario like this? Let's look at the questions you need to ask to determine which technology is best suited to meet the requirements for this particular project.

Do You Need a High Level of Decoupling?

Decoupling different parts of a software solution eliminates mutual dependencies. For example, developing a custom add-on for an SAP application in a different system than an SAP one reduces the risk of the custom code having unintended

[1] See "How to Get the Most Out of ABAP and Java in the Context of SAP Technology: Part 1" by Thomas Weiss and Martin Jaekle in the July-September 2011 issue of *SAPinsider* (sapinsider.wispubs.com).

side effects on the SAP application. Similarly, the custom application is not directly affected by patches and enhancement packages installed in the SAP system. It is also easier to change part of a software solution if it is decoupled from the rest of your software. However, there are also drawbacks of decoupling; for example, you might need additional systems to support the decoupling or a higher data load on the network may decrease performance. To determine whether a decoupled solution is the right approach for you, consider the following questions:

- Does your application use a number of frameworks, modules, etc. from different sources?
- Do you need to be able to exchange different components without much effort?
- Are you willing to put in additional upfront effort and accept the higher TCO and lower performance that can come with a high level of decoupling?

If the answer to any of these questions is "yes," then a decoupled approach is likely right for you. In the case of the example scenario, although the adapted process will access the same SAP ERP back end as the existing process, a high level of decoupling is needed since the insurance company's frequently changing product offerings require constant reassembling or reconfiguration of modules of different granularity.

Do You Need to Integrate Back-End Systems, Applications, and Services?

If your process will need to connect to different systems, take the following into consideration:

- How many systems do you need to connect to?
- What kinds of systems do you need to connect to? For example, do you need connections from an SAP back-end system to non-SAP systems, or do you need to integrate third-party products or libraries?

The higher the number of systems, and the more different systems you need to connect to, the more you can benefit from using an additional system for development and for integrating with other systems. When considering your connectivity options, think about the respective advantages of different connectors in the context of your environment. For instance, Java Connector and .NET Connector work well in high-load scenarios, while SAP NetWeaver Gateway, which supports REST-based services, is suited to low-weight client scenarios. If you need an interface for connecting to a yet-unknown system, web services or enterprise services are a good choice. In considering the different connection types, you will always need to weigh the impact that the type of connectivity will have on your system performance. Even if your current scenario involves only one back-end system, ask yourself these questions:

- Might this change in the near future?
- Is there a chance your company will be part of a merger or an acquisition that will involve different back-end systems?
- Will the growth or strategy of your company require the integration of third-party products or partner solutions so that your solution can cope with its development?

If there is any chance you will have or need additional systems in the future, consider using a Java-based intermediate layer for your connection. This type of middleware provides an additional abstraction layer, which means more overhead in the beginning, but in the long run it translates into faster adaptation to changing requirements. Each new layer may also slow down the performance of your application, but if it is scalable and highly available, such a system can be so efficient that it almost offsets the effect on performance. Of course, there will be a cost to buying or adapting middleware and administrating it, so you have to weigh the trade-off between flexibility and cost.

In the case of the example scenario, there is only one back-end system — the SAP ERP system with SAP for Insurance solutions — but in the future, the insurance company may need to access other systems as well.

Can You Make Do With, or Maybe Even Benefit From, Generated User Interfaces?

Developing a good user interface (UI) from scratch can take quite a bit of time and resources. One of the benefits of using a process modeling solution is that you can take advantage of generated UIs. Developing model-driven UIs is very fast because it requires no coding — you simply define the process model and the UI is generated automatically. If you need to change the UI later, simply adapt the process model and regenerate the UI.

Developing a custom add-on for an SAP application in a different system than the SAP one reduces the risk of the custom code having unintended side effects on the SAP application.

Tip: If needed, you can also scale out or increase the flexibility of your solution later on by using SAP NetWeaver Process Integration or an enterprise service bus.

The downside of generated UIs is that you are bound to the scope of functionality that can be automatically created using the process models. While in principle it is possible to adjust the functionality, manually adapting generated code for UIs is often very time-consuming due to the high complexity of the code. Such code generation usually has to cover a number of possible use cases, so there is likely to be some overhead that is not needed for your particular application, which will increase the size of your application and might have a negative impact on its performance. If you have specific corporate identity requirements that are beyond the scope of generated UIs, you will need to consider a custom solution instead.

In the case of the example scenario, generated UIs work well since there are no complex corporate identity requirements, and especially because the automatic generation of the UIs greatly reduces time to market, which is important when new products with ever shorter life cycles need to be shipped quickly and regularly.

Do Your Business and IT Departments Need to Work on the Same Model?

Quite naturally, business staff and the IT department often have different perspectives on the functionality and applications that a company needs. For example, products are usually designed by business people, who have little or no understanding of the technical implications for the implementation and maintenance of applications that will support their business processes, while the IT department has little knowledge of the plans for new and changing products or the business drivers behind them.

Setting up one common model that both sides can understand, discuss, and work with helps prevent misunderstandings, speeds up planning, and reduces the time from the initial idea to the shipment to market. Using a common model, a product designer models the business process and then discusses it with a developer, who checks the feasibility within the given environment and any technical boundary conditions. After some iteration, the business model is enriched with technical details so that it can be implemented quickly and efficiently, with less risk of errors.

In the case of the example scenario, new products and processes must be developed quickly, and adaptations expected in the future must also have a fast turnaround time. A common model would make the cooperation between business and IT during the development process of the needed solution far more efficient and less error-prone.

Proposed Solution for the Scenario

With all of these questions and their answers in mind, what solution best fits our example scenario? Let's review the requirements:

- We need a high level of decoupling to support frequent product changes.
- A high degree of flexibility is essential, because we expect that several back-end systems, applications, and services will have to be integrated in the near future to accommodate the dynamic growth of the company.
- Generated UIs work well because we don't have complex design needs, and automatic generation speeds time-to-market for new products.
- Due to the fast time-to-market requirement of the new insurance products, it is important for business and IT to work with a shared model to avoid errors and misunderstandings.

The proposed solution is to use SAP NetWeaver Business Process Management (SAP NetWeaver BPM), SAP NetWeaver Composition Environment (SAP NetWeaver CE), and SAP NetWeaver Process Integration (SAP NetWeaver PI), along with an underlying SAP NetWeaver Application Server (SAP NetWeaver AS) Java system. While in principle it is possible to use remote function calls (RFCs) and web services to directly integrate different back-end systems without an intermediate SAP NetWeaver PI system, SAP NetWeaver PI provides a higher level of decoupling.

With SAP NetWeaver CE and SAP NetWeaver BPM, you benefit from the fast model-driven generation of UIs and also from the ability of business staff and the IT department to work together on the same model. If it is set up in a scalable manner using SAP NetWeaver PI, this solution will be high-performing and future-proof.

At first glance, this Java-driven solution might be a surprise since, for the time being, all relevant business logic and data is stored in an ABAP-based SAP ERP system. However, after weighing all of the considerations, it is clear that a solution based on SAP NetWeaver BPM, SAP NetWeaver

CE, and SAP NetWeaver PI provides the most benefit for this particular scenario. This case is a good example of a situation in which your first intuition may not be the best choice for your particular project, and why it is critical to ask and answer questions like the ones outlined here before deciding on a suitable technology.

Example Scenario: Integrating a Partner Solution with Different SAP ERP Systems

It is often the case that your business is only as successful as your software offering's ability to efficiently communicate with partner solutions. For instance, suppose you are an SAP partner developing a Java-based add-on solution for managing ready-mix concrete truck fleets, a time-critical practice because of the material being handled. Your specific Java-based software needs to integrate with the SAP ERP systems of customers, so they can receive ready-mix concrete orders and information and report on the status of these orders. You would also like to have the option to later integrate the solution with non-SAP ERP customer systems. Since this scenario is interactive, real-time status monitoring is expected, and you might want to expand the reach of your solution by offering it on demand.

How do you arrive at the right technology choice for a solution implementation like the one in this example? Let's take a look at the key questions that will show you the way.

First, Revisit the Decoupling and Integration Questions

The first two questions we need to answer for this scenario were covered in the first scenario:

- Do you need a high level of decoupling?
- Do you need to integrate different back-end systems, applications, and services?

In the case of this example, we need a high level of decoupling to support different frameworks, and we need to integrate many different SAP ERP back-end systems, as well as non-SAP ERP systems in the future.

Are Open Standards Important to You?

We also need to gauge the importance of standards. Standards offer a variety of advantages:

- Standards allow an easier exchange between products and vendors.
- There are many existing standards, and they are updated regularly, enabling technology based on these standards to benefit from the latest developments while remaining compliant.
- Standardized products enjoy better market perception and penetration, because compliance with standards protects a customer's investment in the product and reduces dependency on a specific vendor.

Using a standards-based Java interfacing solution helps to ensure a high level of consistency, efficiency, and stability — standards are well established and maintained, especially in the Java world. It is far better to adhere to standards and actively participate in their evolution than to develop proprietary software and interfaces that risk becoming obsolete. Of course, standards usually don't cover the full spectrum of needed functionality, so there are often vendor-specific extensions or value adds. There are also "de facto" standards, such as Eclipse, to consider — although Eclipse is not officially declared as a standard, it is used so widely that compatibility with it should be ensured whenever possible.

It is far better to adhere to standards and actively participate in their evolution than to develop proprietary software and interfaces that risk becoming obsolete.

In the case of the example scenario, we expect to sell the concrete-management solution to many different customers who will use it to connect to the different back-end systems of their own customers. Since it is crucial that the solution is independent of different product versions and partners, we need to use standardized interfaces.

Do You Need High Availability and a High Level of Scalability?

High availability and scalability can be a critical requirement for a solution. Finding a system that is well matched to your specific capacity needs depends on several factors:

- How many concurrent users do you expect?
- How high are the peaks? At which times/intervals do these peaks occur?
- Is multi-tenancy needed for good scalability?
- Will you need or benefit from hardware clusters that can easily be extended?

- Are you willing to design your system for the "worst case," which would lead to a lot of idle capacity under average conditions?

The size of a system depends on the absolute number of concurrent requests, as well as the expected response time. On-demand implementations allow more efficient scaling — servers are included in the cluster during peak times, while for rest of the year they are available for different tasks.

Peak usage levels are also an important factor. For example, think about construction sites during a snowy winter. What happens on days when the temperature suddenly rises above the freezing point? Building crews want to work very fast on these days — peaks that need to be handled quickly and reliably by the system.

Multi-tenancy is another consideration. Since different end customers might use the same application simultaneously, information pertaining to different customers has to be well protected. A good multi-tenancy architecture provides a clear-cut separation between different customers' critical information while common data is held in a shared cache to provide a better scalability. It is also important to consider the mid-term and long-term requirements of your processes, and to consider where the processing load can be handled in the most efficient way.

In the end, only you can make the right choice for your project — whether it is ABAP, Java, or a mixed solution. Remember that the right decision depends on the particular conditions and requirements of your project, and that different criteria can overwrite another.

In the example scenario, a secure, highly available system with reliable failover mechanisms is essential. Even, if one server fails, the processing of the information must continue seamlessly on another server in the cluster. These considerations are focused on the (web) front end and the intermediate layer, both of which are usually well served by Java-based implementations. If scalability and high throughput is needed in the back end, on the other hand, ABAP has considerable strengths.

Proposed Solution for the Scenario

The requirements for our example scenario provide us with a map to the solution of choice:

- We need a high level of decoupling, since we might substitute the middleware someday.
- We need to integrate different systems and applications — that is, we need to integrate the Java solution for concrete management with customers' ABAP-based SAP ERP back ends (and with customers' non-SAP ERP back-end systems in the future).
- We need open standards in order to offer an interface to which any customer can connect.
- We need high levels of scalability and availability to ensure reliability.

The proposed solution for this scenario is SAP NetWeaver PI with an underlying SAP NetWeaver AS Java system. An intermediate hub such as SAP NetWeaver PI supports strong decoupling, and because it supports standards such as web services, it enables the Java solution to integrate with different back-end systems, whether they are SAP ERP systems or ERP systems from other vendors. To meet capacity needs, SAP NetWeaver PI can be set up in a scalable cluster, and it also offers reliable messaging, along with monitoring tools so that you can easily track messages.

Summary

This two-part article series has guided you through a number of questions to ask before deciding whether to use ABAP or Java for your development project. Some answers might motivate you to consider an ABAP-based solution, while other considerations may convince you to develop a Java-driven solution. Remember that the right decision depends on the particular conditions and requirements of your project, and that different criteria can overwrite another. You may have many good reasons to implement an add-on solution using a particular technology, but if personnel and performance costs are too steep, for example, or if an upcoming change such as a new acquisition requires a different solution, you will need to at least rethink the decision.

In the end, only you can make the right choice for your project — whether it is ABAP, Java, or a mixed solution. Regardless of whether you implement a solution in ABAP or prefer model-driven development with SAP NetWeaver BPM or an SAP NetWeaver PI-supported integration scenario, you can count on the same level of security, support, and comprehensive lifecycle management you are used to relying on from SAP. ■