BPM in a Data Driven Eco System

Austin Rosenfeld

BPM systems are for managing processes, not data. But processes need data like animals need water, so understanding the symbiosis between the storage system for process state, history, and rules and the ultimate resting place for the data collected by those processes is an important part of system architecture. In fact, the relationship between process and data is bidirectional, so a well designed system accounts for both the data processes needed to operate and the data they collect and calculate. While the BPM engine does the heavy lifting with task routing and event sequencing, various repositories and content management systems throughout the enterprise have roles to play in the process-data ecosystem as well.

Where does data come from? Simple systems, like an expense report processor, have very few dependencies on external data. For the most part, the process initiator creates all the data the process needs by entering in what was purchased and how much it cost on the first screen; the rest of the process operates on this data in a vacuum. As processes increase in complexity, their data dependencies also increase. A process managing customer interactions, for example, may depend on customer information from the enterprise resource planning (ERP) system, department information from the company registry (LDAP, Active Directory, etc.), and even transaction information from the eCommerce site. When the process is complete and all of the approvals are checked off, data updates may even be written back to these sources, since the ERP system is the system of record for invoices and the process might occasionally need to change one.

Writing data back is the easy part: that is just an integration, which has been done since the stone age of BPM (2005), when major BPM suites supported a "Write to ERP" node. Surfacing the data that originates in ERP through the BPM engine and into the portal interface, though, is a more difficult challenge. If we want to allow the process user to browse a list of all invoices currently housed in ERP, filter them by customer, sort them, etc., we are doing more than just a simple integration. The old approach was to pull all of these records into the BPM engine and expose a copy of them, but this design does not scale. It may work for queries such as "show me a dropdown of our 200 customers," but it doesn't work for "show me the 10,000 web transactions from this customer," or "show me all the invoices for all my customers, because I want to fish for one."

The solution to this data dilemma is for BPM suites to evolve. To be a first-class player in the enterprise stack, the BPM suite needs to provide a seamless interface to data that is housed externally. We have already seen the transition to BPM suites supporting externalized integrations by playing nicely with enterprise service buses (ESBs). As systems become more complex and depend on more enterprise resources that are stored in their own systems of record, it is time to treat data the same way. More importantly, the mechanism for this data-oriented integration needs to stay true to BPM's DNA: expose a set of "Legos" from which designers can build applications; do not force them to write custom code.

For a subset of the problem, we can string together a "Get Data from ERP" node followed by a task screen, but that only scratches the surface of the external data problem. What if the user completing the task chooses a customer from the list, but the process dashboard needs to display detailed information about that customer? Dashboards are passive and get stale easily. If customers change their names through mergers and acquisitions during the lifespan of a year-long process, how are the dashboards for all of the processes related to those customers updated? The answer should be that each dashboard view is a pull of the data from the corresponding system of record, not that the data is cached in the BPM engine at the time of the initial selection of that customer. Again, the BPM tool should play nicely with the system of

record. There should be a smooth interface for the process designer to configure that dashboard to show the external data, with no code.

BPM suites have long been central to the technical enterprise stack, delegating to ESBs for integration, content management systems for document storage, and credential stores for authentication. It is time for them to delegate data management appropriately as well. Demand more from your BPM tool provider. The ones we use on our projects support this modern view of data ownership. Yours should too.

Author

Austin Rosenfeld is the founder of Macedon Technologies, formerly known as Macedon Consulting. Previously, he was a product architect at Appian and ran the BPM consulting practice at Amentra. For more information on how BPM suites can deliver flexible solutions for your organization, contact austin.rosenfeld@macedontechnologies.com.

BPTrends Linkedin Discussion Group

We created a BPTrends Discussion Group on Linkedin to allow our members, readers and friends to freely exchange ideas on a wide variety of BPM related topics. We encourage you to initiate a new discussion on this publication, or on other BPM related topics of interest to you, or to contribute to existing discussions. Go to Linkedin and join the **BPTrends Discussion Group.**

