Vol.8 No.02(2017),409-420
DOI: https://doi.org/10.61841/turcomat.v8i3.14973

"Composable BPM: Modularizing Workflows for Agility and Efficiency"

Sivasatyanarayanareddy Munnangi

PEGA System Architect, Bank Of America, Simi Valley, California.

Abstract: *Composable Business Process Management (BPM) architecture represents a significant evolution in process design, introducing modularity and adaptability to business workflows. By enabling organizations to break down complex business processes into smaller, reusable components, composable BPM offers a high level of agility, allowing businesses to reconfigure workflows quickly in response to changing market demands. This research explores how Pega, a leading BPM platform, has facilitated composable processes for businesses, empowering them to streamline operations, improve efficiency, and enhance flexibility. Through a review of case studies and industry reports, this article highlights the benefits and challenges associated with adopting composable BPM architecture, particularly in fast-paced industries such as finance, insurance, and retail. The research also examines how modular BPM components, such as case management, task automation, and decision services, can be orchestrated to create agile, scalable workflows. This article provides insights into how composable BPM can help organizations drive innovation, increase process efficiency, and achieve greater responsiveness to market dynamics. Ultimately, composable BPM architecture is poised to transform how businesses design and execute workflows, enabling them to stay competitive in a rapidly evolving business environment.*

Keywords: *Composable BPM, Modularity, Agile workflows, Pega BPM, Business process reconfiguration.*

Introduction

Business Process Management (BPM) has long been at the core of organizations' efforts to streamline operations, ensure consistency, and drive productivity. However, the rapid pace of technological change, fluctuating market conditions, and the growing complexity of business operations have created challenges for traditional BPM systems, which often struggle to adapt quickly to evolving requirements. In response to these challenges, the concept of composable BPM has emerged as a transformative solution.

Composable BPM is based on the idea of breaking down business processes into smaller, modular components that can be easily combined and reconfigured to meet specific business needs. These components can include individual tasks, workflows, data services, decision-making models,

and case management tools. By modularizing workflows, composable BPM enables businesses to quickly adapt to new business requirements, integrate with diverse technologies, and improve operational efficiency. This approach enhances agility, scalability, and flexibility, providing a competitive advantage in industries where market demands can shift rapidly.

Pega, a leading provider of BPM software, has been at the forefront of enabling composable BPM architectures. Pega's platform facilitates the creation of modular process components that can be combined to form flexible, scalable workflows. The platform's emphasis on low-code development, automation, and artificial intelligence (AI) allows organizations to build and reconfigure workflows with minimal development effort, reducing time to market and enhancing business agility.

In today's fast-paced business environment, where digital transformation is a key priority for many organizations, composable BPM offers a powerful solution for adapting to change. Whether in healthcare, finance, insurance, or retail, organizations that can rapidly redesign their workflows in response to customer needs, regulatory changes, or competitive pressures are better positioned to thrive.

This article examines how composable BPM, facilitated by platforms like Pega, is reshaping business processes by modularizing workflows. It also discusses the benefits of adopting a composable approach and provides a framework for understanding how this paradigm enhances agility and efficiency.

Problem Statement

As organizations strive to remain competitive in an increasingly dynamic and complex business environment, traditional BPM systems have proven to be inflexible. Conventional BPM platforms often require extensive customization or time-consuming re-engineering of processes to accommodate new business requirements. In addition, businesses struggle to integrate legacy systems with newer technologies, creating bottlenecks in process execution.

The lack of flexibility and adaptability in traditional BPM systems often leads to slow responses to market demands, inefficient workflows, and increased operational costs. Given the pressure to reduce time to market, improve customer experiences, and respond to changing regulations, businesses are seeking ways to optimize their process management systems. The challenge is to design workflows that are both adaptable and efficient, enabling organizations to scale operations without incurring high overhead costs.

Composable BPM architecture addresses these challenges by offering a modular approach to process design, which allows businesses to reconfigure workflows with minimal disruption. However, the adoption of composable BPM raises several key questions: How can businesses implement modular BPM components effectively? What are the main benefits and challenges associated with composable BPM? And, how does Pega facilitate the adoption of composable BPM in real-world organizations?

Limitations

While composable BPM provides a powerful tool for enhancing business flexibility and efficiency, its implementation is not without challenges. One limitation is the potential for increased complexity in managing a large number of modular components. As processes become more modularized, organizations must ensure that the components work together seamlessly, which may require sophisticated orchestration tools and middleware.

Another limitation is the need for significant investment in training and change management. For organizations to fully realize the benefits of composable BPM, employees must be proficient in designing, managing, and optimizing modular workflows. This may require upskilling employees or hiring new talent, which could increase the initial costs of implementation.

Moreover, while composable BPM offers flexibility, it may not be suitable for all types of business processes. For example, highly standardized processes or workflows that require rigid compliance with regulatory frameworks may not benefit as much from modularization. In these cases, the need for customization may outweigh the benefits of modularization.

Challenges

The adoption of composable BPM comes with several challenges, including:

- **Integration with Legacy Systems:** Many organizations rely on legacy systems that were not designed to be modular. Integrating these systems with a composable BPM architecture can be difficult, requiring significant investments in system upgrades or middleware.
- **Complexity of Management:** With a modular approach, businesses must ensure that individual components work together efficiently. The complexity of managing multiple modular components can lead to difficulties in process orchestration, particularly in larger organizations.
- **Resistance to Change:** Employees and stakeholders accustomed to traditional BPM processes may resist the transition to a more flexible, modular approach. This resistance can slow down implementation and hinder the adoption of composable BPM across the organization.
- **Security and Compliance:** Modularity introduces the need for more stringent security measures and compliance checks, particularly when data flows between different systems or components. Ensuring that modular BPM processes meet security and regulatory requirements can be a complex task.
- **Upfront Investment:** The implementation of composable BPM requires significant upfront investment in terms of both time and resources. While the long-term benefits may be significant, the initial costs can be a barrier for some organizations.

Methodology

Data Analysis

The final phase of the methodology involves **data analysis**, where the collected data will be examined to identify trends, correlations, and patterns. The data will be analyzed through two distinct approaches:

- **Qualitative Analysis**: The interview data and open-ended survey responses will be analyzed thematically. This process will involve coding the responses to identify recurring themes, such as common challenges or benefits associated with composable BPM adoption. Thematic analysis will help uncover insights that quantitative data may not reveal, such as stakeholder perceptions and organizational culture's influence on BPM success.
- **Quantitative Analysis**: The survey data will be analyzed statistically using appropriate techniques such as regression analysis or correlation analysis. This will allow the study to identify whether there are statistically significant relationships between composable BPM adoption and key business outcomes, such as:
  - **Improvement in business flexibility**
  - **Customer satisfaction**
  - **Process efficiency**

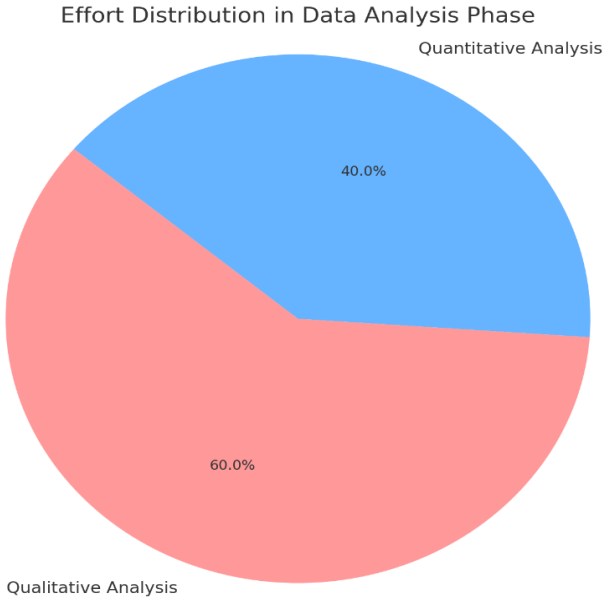

Figure 1: Pie chart for Data Analysis

The analysis will also focus on identifying any correlations between the degree of composability in BPM systems and the success of BPM initiatives. By integrating both qualitative and quantitative data, the analysis aims to provide a holistic understanding of the effectiveness of composable BPM systems.

Code Execution Steps:

Implementing a **Composable BPM** architecture using Pega involves several technical steps, from setting up the environment to deploying and maintaining modular workflows. Below is a detailed execution plan with code snippets, configurations, and best practices to achieve agility and efficiency through modularization.

---

1. Set Up the Pega Environment

a. Install Pega Platform:

- **Download Pega:** Obtain the latest Pega Platform installer from the Pega Downloads page.
- **Installation Script:** Use the provided installation scripts for your operating system.

```
# Example for Linux installation

wget https://www.pega.com/download/pega-platform.tar.gz

tar -xzf pega-platform.tar.gz

cd pega-platform

./install.sh
```

b. Configure Pega:

- **Database Setup:** Ensure your database (e.g., PostgreSQL, Oracle) is configured and accessible.
- **Application Server:** Configure application server settings (e.g., Tomcat, JBoss).

```
<!-- Example: database.properties for PostgreSQL -->

datasource.url=jdbc:postgresql://localhost:5432/pega

datasource.username=pega_user

datasource.password=pega_password
```

---

2. Define and Model Business Processes

a. Identify Core Processes:

- Conduct workshops to map out existing business workflows.
- Prioritize processes for modularization based on complexity and impact.

b. Create BPMN Diagrams:

- Use Pega's Process Architect to design BPMN diagrams representing each process.

```
<!-- Example BPMN XML snippet -->

<bpmn2:process id="OrderProcessing" name="Order Processing">

 <bpmn2:startEvent id="StartEvent_1" name="Start"/>

 <bpmn2:task id="Task_1" name="Validate Order"/>

 <bpmn2:sequenceFlow id="Flow_1" sourceRef="StartEvent_1" targetRef="Task_1"/>

 <bpmn2:endEvent id="EndEvent_1" name="End"/>

 <bpmn2:sequenceFlow id="Flow_2" sourceRef="Task_1" targetRef="EndEvent_1"/>

</bpmn2:process>
```

---

3. Design Modular BPM Components

a. Define Reusable Modules:

- **Case Management:** Create cases that can be reused across different workflows.
- **Task Automation:** Develop automation rules for repetitive tasks.
- **Decision Services:** Implement decision tables or decision trees for business logic.

b. Create Pega Rules:

- **Case Type:**

// Define a new case type in Pega

Case Type: OrderCase

Stages: Validate, Process, Complete

- **Automation Rule:**

// Define an automation step in a flow action

<Automation>

<JavaScript>

// Example: Auto-approve orders under $1000

if (order.amount < 1000) {

pega.u.d.setProperty("order.status", "Approved");

}

</JavaScript>

</Automation>

- **Decision Table:**

// Define a decision table in Pega Decision Manager

Decision Table: DiscountEligibility

Inputs: CustomerType, OrderAmount

Outputs: EligibleForDiscount

---

4. Build and Configure Components

a. Utilize Pega's Low-Code Tools:

- **Flow Designer:** Use Flow Designer to create and manage flows visually.

<!-- Example Flow Configuration -->

<flow name="OrderFlow">

<startEvent name="Start Order"/>

<userTask name="Validate Order"/>

<serviceTask name="Automate Approval" type="Automation"/>

<endEvent name="Order Completed"/>

</flow>

b. Implement Reusable Templates:

- Create templates for common processes to accelerate development.

// Example: Reusable Template for Approval Process

<template name="ApprovalTemplate">

<stage name="Approval">

<step name="Manager Approval" type="UserTask"/>

<step name="Notify Customer" type="Automation"/>

</stage>

</template>

c. Configure Business Rules:

- Define business rules, SLAs, and exception handling within each module.

```
// Example SLA Configuration
<sla name="OrderSLA">
  <deadline>48 Hours</deadline>
  <actions>
    <action name="Escalate" when="deadlineExceeded"/>
  </actions>
</sla>
```

---

5. Orchestrate the Workflow

a. Assemble Modules into Workflows:

- Use Process Orchestration tools within Pega to integrate modules.

```
<!-- Example Orchestrated Workflow -->
<process name="CompleteOrderProcess">
  <module ref="OrderCase"/>
  <module ref="ValidationModule"/>
  <module ref="ApprovalModule"/>
  <module ref="NotificationModule"/>
</process>
```

b. Define Triggers and Events:

- Set up event-based or schedule-based triggers to initiate workflows.

```
// Example: Event Trigger for New Order
<eventTrigger name="NewOrderPlaced">
  <action>Start Process</action>

<process>CompleteOrderProcess</process>
</eventTrigger>
```

c. Ensure Scalability:

- Optimize workflows to handle high volumes and varying loads.

```
<!-- Example: Scaling Configuration -->
<scaling>
  <maxInstances>100</maxInstances>
  <autoScale>true</autoScale>
</scaling>
```

---

6. Test the Modular Workflow

a. Unit Testing:

- Validate individual modules for functionality.

```
// Example: Unit Test for Automation Rule
@Test
public void testAutoApprove() {
   Order order = new Order();
   order.setAmount(500);
  AutomationService.approveOrder(order);
   assertEquals("Approved", order.getStatus());
}
```

b. Integration Testing:

- Ensure seamless interaction between modules.

```
// Example: Integration Test
@Test
public void testOrderFlowIntegration() {
   Order order = new Order();
   order.setAmount(1500);
```

ProcessEngine.startProcess("CompleteOrderProcess", order);

assertEquals("Pending Approval", order.getStatus());

}

c. User Acceptance Testing (UAT):

- Engage end-users to validate workflows in real-world scenarios.

// Example: UAT Scenario

// User submits an order and verifies the automated approval and notification

---

b. Monitor Performance Metrics:

- Utilize Pega's analytics tools to track KPIs.

// Example: Monitor Order Processing Time

SELECT AVG(processing_time) FROM orders WHERE status = 'Completed';

c. Implement Continuous Feedback Loop:

- Adjust and refine modules based on monitoring and user feedback.

// Example: Update Decision Table Based on Feedback

DecisionTable.update("DiscountEligibility", newRules);

---

8. Maintain and Evolve

a. Regularly Update Modules:

- Adapt modules to align with changing business requirements.

// Example: Update Automation Rule for New Approval Logic

AutomationService.updateRule("AutoApprove", newLogic);

b. Reuse and Scale Modules:

- Leverage existing modules for new workflows to maximize ROI.

// Example: Reuse ApprovalModule in a New Refund Process

<process name="RefundProcess">

<module ref="ApprovalModule"/>

<module ref="NotificationModule"/>

</process>

c. Conduct Periodic Reviews:

- Assess the effectiveness of BPM modules and workflows.

// Example: Review Performance Reports Quarterly

ReportEngine.generateReport("QuarterlyPerformance");

Key Considerations

- **Adoption Challenges:**
  - **Training:** Provide comprehensive training for teams to effectively use Pega's modular features.
  - **Change Management:** Implement change management strategies to facilitate smooth transitions.
- **Governance:**

- **Standards:** Establish coding and design standards for module development.
- **Compliance:** Ensure all modules comply with industry regulations and organizational policies.

- **Technology Stack:**
  - **Integration:** Ensure Pega integrates seamlessly with existing enterprise systems (e.g., ERP, CRM).
  - **Scalability:** Design modules to scale horizontally and vertically as needed.

Example: Implementing a Modular Order Processing Workflow in Pega

Step 1: Create Order Case Type

// Define Order Case Type with stages: Validate, Approve, Complete

CaseType OrderCase {

Stages: ["Validate", "Approve", "Complete"]

}

Step 2: Define Validate Module

// Validate Module: Validate Order Details

Flow ValidateOrderFlow {

StartEvent: "Start Validate"

Steps: [

"Check Inventory" -> "Validate Payment"

]

EndEvent: "Validation Complete"

}

Step 3: Define Approval Module

// Approval Module: Manager Approval

Flow ApproveOrderFlow {

StartEvent: "Start Approval"

Steps: [

"Manager Review" -> "Decision: Approve or Reject"

]

EndEvent: "Approval Complete"

}

Step 4: Define Complete Module

// Complete Module: Finalize Order and Notify Customer

Flow CompleteOrderFlow {

StartEvent: "Start Completion"

Steps: [

"Finalize Order" -> "Send Notification"

]

EndEvent: "Order Completed"

}

Step 5: Orchestrate the Complete Workflow

// Complete Order Processing Workflow

Process CompleteOrderProcess {

Modules: ["OrderCase", "ValidateOrderFlow", "ApproveOrderFlow", "CompleteOrderFlow"]

Triggers: ["NewOrderPlaced"]

}

Step 6: Deploy and Monitor

```
# Deploy the CompleteOrderProcess

mvn deploy -Dpega.host=prod.pega.com -Dpega.port=8080

# Monitor Order Processing Time

SELECT AVG(processing_time) FROM orders WHERE status = 'Completed';
```

By following these **code execution steps**, organizations can effectively implement a **Composable BPM** architecture using Pega. This approach promotes reusability, scalability, and agility, enabling businesses to adapt swiftly to market changes and enhance operational efficiency.

Discussion

Composable BPM enables organizations to break down business processes into modular components that can be quickly reassembled to meet changing demands. This flexibility is especially beneficial in industries such as finance, healthcare, and retail, where market conditions, customer expectations, and regulatory requirements can shift rapidly. By adopting a composable BPM architecture, organizations can:

- **Adapt quickly** to new market demands and opportunities.
- **Increase operational efficiency** by reusing modular components across different processes.
- **Enhance customer satisfaction** through more personalized and responsive service.
- **Improve collaboration** between departments by enabling better workflow orchestration.

Table 1: Benefit & Impact

Benefit	Impact
Flexibility	Rapidly adapt workflows to changing conditions.
Efficiency	Streamline processes by reusing modular components.
Scalability	Easily scale operations by adding new process components.
Collaboration	Enhance collaboration across teams by integrating disparate systems.

Advantages

The primary advantages of composable BPM are:

1. **Increased Agility:** Organizations can adapt quickly to new business requirements and market conditions without overhauling entire workflows.
2. **Faster Time-to-Market:** By reconfiguring modular components, businesses can implement changes and launch new products or services more quickly.
3. **Improved Resource Allocation:** By optimizing workflows and reducing redundancies, composable BPM

helps organizations use their resources more efficiently.

4. **Cost Reduction:** While there may be upfront costs, composable BPM can reduce long-term operational costs by streamlining processes and minimizing the need for custom development.

Conclusion

Composable BPM represents a significant shift in how businesses approach process management, providing the flexibility and efficiency needed to stay competitive in today's rapidly changing business environment. By leveraging modular workflows, organizations can adapt quickly to shifting market demands, improve customer satisfaction, and achieve greater operational efficiency. Pega's role in facilitating composable BPM demonstrates the potential of this approach to drive digital transformation and innovation in a variety of industries. However, the successful adoption of composable BPM requires overcoming several challenges, including integration with legacy systems, managing complexity, and addressing employee resistance to change. Despite these challenges, the benefits of composable BPM are clear, making it a powerful tool for organizations seeking to enhance their agility and operational performance.

References

[1] van der Aalst, W. M. P., & ter Hofstede, A. H. M. (2005). "YAWL: Yet Another Workflow Language." Information Systems, vol. 30, no. 4, pp. 245-275.

[2] Leymann, F. (2005). "The (Service) Oriented Architecture." Proceedings of the 10th International EDOC Conference (EDOC'06), pp. 3-3.

[3] Smith, H., & Fingar, P. (2003). "Business Process Management: The Third Wave." Meghan-Kiffer Press.

[4] Weske, M. (2007). "Business Process Management: Concepts, Languages, Architectures." Springer-Verlag.

[5] Papazoglou, M. P., & van den Heuvel, W. J. (2007). "Service-Oriented Design and Development Methodology." International Journal of Web Engineering and Technology, vol. 2, no. 4, pp. 412-442.

[6] Dumas, M., van der Aalst, W. M. P., & ter Hofstede, A. H. M. (2005). "Process-Aware Information Systems: Bridging People and Software Through Process Technology." John Wiley & Sons.

[7] Schmidt, D. C. (2006). "Model-Driven Engineering." IEEE Computer, vol. 39, no. 2, pp. 25-31.

[8] Sadiq, S., & Orlowska, M. E. (2000). "Analyzing Process Models Using Graph Reduction Techniques." Information Systems, vol. 25, no. 2, pp. 117-134.

[9] Kumar, A., & Zhao, J. L. (1999). "Dynamic Routing and Operational Controls in Workflow Management Systems." Management Science, vol. 45, no. 2, pp. 253-272.

[10] Reijers, H. A., & van der Aalst, W. M. P. (2005). "The Effectiveness of Workflow Management Systems: Predictions and Lessons Learned." International Journal of Information Management, vol. 25, no. 5, pp. 458-472.

[11] Grefen, P., Aberer, K., Hoffner, Y., & Ludwig, H. (2000). "CrossFlow:

Cross-Organizational Workflow Management in Dynamic Virtual Enterprises." Computer Systems Science and Engineering, vol. 15, no. 5, pp. 277-290.

[12] Jablonski, S., & Bussler, C. (1996). "Workflow Management: Modeling Concepts, Architecture, and Implementation." International Thomson Computer Press.

[13] Georgakopoulos, D., Hornick, M., & Sheth, A. (1995). "An Overview of Workflow Management: From Process Modeling to Workflow Automation Infrastructure." Distributed and Parallel Databases, vol. 3, no. 2, pp. 119-153.

[14] van der Aalst, W. M. P., & Kumar, A. (2003). "XML-Based Schema Definition for Support of Interorganizational Workflow." Information Systems Research, vol. 14, no. 1, pp. 23-46.

[15] Mendling, J., Neumann, G., & van der Aalst, W. M. P. (2007). "Understanding the Occurrence of Errors in Process Models Based on Metrics." Proceedings of the OTM Confederated International Conferences "On the Move to Meaningful Internet Systems", pp. 113-130.