The 2006 BPMS Report:

Lombardi TeamWorks v5.5

Bruce Silver

www.brsilver.com/wordpress
1. Vendor and Product Overview

Lombardi Software is a private company focused exclusively on Business Process Management software. Lombardi designs, sells, and supports process improvement solutions for Fortune 1000 organizations in industries ranging from financial services to manufacturing to energy and telecommunications. Based in Austin, Texas, the company was founded in 1998, and has approximately 120 employees.

Lombardi’s TeamWorks software focuses on “operational processes,” meaning complex flows spanning organizational and system boundaries, characterized by four distinct attributes:

- Processes with impact on overall business performance, such as inventory, DSO, cycle-time, customer retention – strategic business processes affecting the bottom line
- Processes subject to continual change
- Processes requiring monitoring and management of exceptional events
- Processes requiring real-time visibility and optimization of system and user performance

Lombardi is known for its strength in human interaction in some form of collaborative case management or exception process, such as dispute resolution, product returns, tax reconciliation, loan origination, or supply chain management. TeamWorks is ideally suited for processes that change dynamically based on business factors.

A sampling of Lombardi customers and applications includes:

- Aflac, a leading insurance provider in the US and abroad, insuring over 40 million people worldwide. With a large invoice volume, Aflac's Invoice Reconciliation team uses TeamWorks to resolve disputes within their 30-day cycle, with visibility into process performance. They are able to automatically prioritize reconciliation tasks based on time and value, which means smaller claims can go straight through to resolution, freeing time for employees to focus on critical disputes. Notifications are sent when a dispute has aged 20 days without resolution. Aflac is expanding their TeamWorks implementation into New Application Processing and Policy Administration departments.

- Dell, a world leader in computer systems. Dell uses TeamWorks to manage distressed shipments, products scheduled for delivery that, for any number of reasons, could not be delivered to the customer. TeamWorks manages the process linking customer care representatives, the carrier, and the customer to resolve these distressed shipments, saving Dell millions per quarter. Action can be taken even before the shipment arrives back at the company, saving hundreds of dollars per shipment and greatly improving the customer experience.

- Pulte Mortgage, provider of loan services to 85% of Pulte Home buyers. Pulte uses TeamWorks to manage their end-to-end loan origination processes, giving users the ability to view a loan application at any step in the process, remove bottlenecks, and continuously improve process performance. Besides enhancing customer service, TeamWorks significantly lowers operational costs.
Lombardi TeamWorks Enterprise Edition v5.5 is a complete BPM Suite supporting modeling and simulation analysis, human workflow, application integration, business rules, and real-time performance monitoring and optimization. Components of the suite include:

- **TeamWorks Authoring Environment**, a collaborative Eclipse-based tool shared by business and IT, providing both analytical modeling and executable design. TeamWorks is one of the only BPMS offerings in which both modeling and design leverage the BPMN standard, including events and other advanced process semantics.

- **TeamWorks Process Server**, the J2EE-based runtime engine for process execution.

- **TeamWorks Performance Server**, a parallel engine for business activity monitoring, reporting, and performance optimization.

- **TeamWorks Shared Model**, a common fixed-schema data model shared by Performance Server and Process Server, unifying data definitions for all Process and Performance elements and providing a common basis for historical, in-flight, and simulated execution data.


- **TeamWorks Process Optimizer**, a unique business analyst tool combining the usually separate functions of simulation analysis and activity monitoring, supporting configurable Business Scenarios combining historical, in-flight and what-if analysis, with optimization recommendations provided by the tool.

**Lombardi for Office 2003** is a new offering, engineered for simple deployment and access to tasks, performance metrics, and other BPM artifacts through Microsoft Outlook, Word, and Excel rather than from a web portal. Lombardi for Office also supports InfoPath forms, .Net APIs, System Management Server, Active Directory Group Policy Objects, and other Microsoft platform components and services, simplifying enterprise deployment in Microsoft shops.

**Lombardi for Partners** is the generic name for the OEM edition of TeamWorks. It essentially contains all of TeamWorks Enterprise, but supports installation customized to the partner environment. Lombardi for Partners is used by Cognos, Paisley Consulting (compliance), FiServ Insurance Solutions, and Extensity.

2. Environment and Architecture

2.1 **Architecture Overview**

Figure 1 illustrates the TeamWorks functional architecture:

The **Process Server** is responsible for managing and executing process model definitions and in-flight instances. This server provides workflow facilities, such as task management as routing, as well as rule evaluation, simulation, and other capabilities. The **Process Components** are stored and shared as XML in the process repository within the RDMBS.

TeamWorks provides extensive levels of caching to optimize run-time performance. Process execution is dynamic, evaluating each step and transition within the process diagram in real time. The Process Server also briefly handles reporting data before it off-loads it to the Performance Server at each execution of a Tracking Point (see Section 10). In addition to Process Server’s internal execution capabilities, TeamWorks also provides several frameworks that enable integration, event monitoring, and security.
Figure 1. TeamWorks functional architecture. Source: Lombardi

**TeamWorks Connector Framework** provides connectivity to external systems, databases, and Web services. TeamWorks connectors can be implemented using remote client API libraries, SOAP Web service protocol, or message-based connectors that implement the JCA standards. Application-specific connectors are available in TeamWorks and through a partnership with iWay, a third-party provider of native data, application and integration connectors. Integration definitions created in TeamWorks are stored in the Process Library and can be used by any process. Additionally, any process or subprocess in TeamWorks can be exposed as a Web service with a click of the mouse, enabling process reuse through a service-oriented architecture.

The **Event Manager** is responsible for time- and event-based process activities. Events may be derived from TeamWorks’ executable processes and workflow, or events may originate from external applications, such as existing CRM, ERP, or workflow applications. Events received via the **TeamWorks Messaging Framework** from external services via JMS, HTTP, or SOAP are automatically correlated to the appropriate business process instances using configurable matching logic. Internal event examples include Timer Events, used to define escalation rules within the process, as well as Exception Events, or paths that provide the mechanisms for “throwing” and “catching” process exceptions (which are most likely asynchronous). All events, triggers, and deadlines that govern process management can be modeled directly in the process diagram in TeamWorks.

TeamWorks externalizes all of its security and authentication in an open architecture based on the Java Authentication and Authorization Service (JAAS) standards. Security may be managed by the default TeamWorks provider, by the application server, an LDAP server, or a mix of security providers. Direct support for products like Microsoft Active Directory and other LDAP servers simplify user management. No maintenance of TeamWorks is required when users are added, deleted, and modified in these systems, and support for single sign-on products like Netegrity allows customers to leverage their enterprise authentication investments.
**TeamWorks Performance Server** collects performance data representing key business events and metrics as processes are being executed. Performance management is completely model-driven, using the process model to define the business events and metrics, event correlation, and aggregation of the performance data. During execution, Tracking Points in the process model collect performance data and sends it to the Performance Server, which correlates business events in real time and aggregates performance data into a single database view for reporting and auditing in TeamWorks ScoreBoards.

![Diagram](image)

**Figure 2. Sample Physical Architecture. Source: Lombardi**

Tracked data is stored in the Performance Server using a fixed physical schema. Since the underlying schema never changes, performance tracking can be added or changed at any time.
without requiring underlying database changes or adversely affecting existing reports. This greatly simplifies deployment of new performance tracking.

2.2 Runtime Architecture

TeamWorks’ was developed using Java, J2EE and XML in order to ensure easy integration with existing software infrastructure. The TeamWorks application runs inside a J2EE-compliant Application Server through a Java Virtual Machine (JVM). The Application Server handles basic connections for TeamWorks, the containers for TeamWorks, as well as the underlying security and permissions. In addition, TeamWorks requires a JDBC-compliant database. TeamWorks runs on Unix, Linux, or Windows operating systems using either JBOSS, WebLogic or WebSphere J2EE application servers.

TeamWorks Servers may be partitioned for autonomy, or load-balanced and clustered for increased scalability and availability. High scalability is achieved through clustering at each layer of the application. The Web Servers can be clustered to support a large user community. Application Servers can be clustered for high volume and a large user community. The DBMS layer can be replicated, mirrored, or clustered for high scalability and availability. Figure 2 shows a basic Physical Topology for a TeamWorks deployment.

Taking advantage of the data-driven architecture, TeamWorks services are largely stateless, which ensures linear scaling, meaning that adding a server will add a specific amount of capacity, which is important to lowering the total cost of ownership of large applications.

2.3 Authoring Environment

The TeamWorks Authoring Environment is a comprehensive Eclipse-based modeling and design environment shared by business and IT. It includes several tools, each with its own Eclipse perspective:

The Process Modeler perspective (Figure 3) lets business analysts create business process diagrams (BPDs) using standard BPMN notation and semantics. Process Modeler supports two alternative image themes for the BPMN constructs, one using “friendly” Lombardi-specific icons, and another that uses the BPMN shapes straight from the OMG spec (Figure 4). BPDs created in Process Modeler are stored in the TeamWorks Library, enabling collaboration and reuse across the enterprise, and comprehensive process documentation can be generated from the Process Modeler perspective at any time.

The Service Modeler perspective (Figure 5) allows design of Teamworks Services, which is Lombardi’s term for the implementation of process activities, such as user tasks, integration actions, scripts, or business rules. In TeamWorks, a single service is frequently composed of multiple steps, designed in a flowchart similar to a BPD. A typical example is a multistep user task, what other BPMSs call a screenflow, composed of multiple web pages or forms linked by database lookups or other automated functions.

The Service Modeler perspective provides several tools used to design specific process components. These include:

- Coach Designer – build web forms used in task user interface
- Rule Designer – design and manage business rules
- Integration Wizard – create integration components by point-click introspection of Java classes and web services
- Report Wizard – build complex reports without programming
Figure 3. Process Modeler perspective. Source: Lombardi

Figure 4. Business Process Diagram using BPMN image theme. Source: Lombardi
The Process Optimizer perspective (Figure 6) lets analysts model and monitor the typical duration, frequency, and resource consumption of each component in a process. It also provides recommendations for resolving process bottlenecks. Using defined performance goals, simulated performance data can be compared against target data.

The Process Inspector perspective supports validation and graphical debugging of process designs and tracking in-flight process execution. Process designers can step through models using playbacks, and inspect, troubleshoot, and correct problems in running processes and services.
3. Analytical Modeling

3.1 Modeling

The Process Modeler perspective allows business analysts to describe and document the sequence and performers of process activities without needing to know how activities are implemented. In that sense, Process Modeler can be described as “modeling” and Service Modeler as “implementation design.” Using BPMN, the basic activity flow in Process Modeler is laid out in swimlanes, each swimlane representing a process participant, either a user role or the TeamWorks system. For simulation purposes, each participant is modeled with a unit hourly cost and a “capacity” for the process, i.e., how many units of that resource are available in the swimlane (Figure 8). Modeling generates detailed documentation about the Business Process Definition in HTML (Figure 9). The look and feel can be customized by XSLT stylesheets.

Performance modeling is also an important aspect of Process Modeler. Tracking Events inserted in the flow define points where snapshots of data are captured for aggregation in performance metrics. In this way, the performance metrics and the business events related to them can be specified independently and in advance of any implementation design in Service Modeler. See Section 10 for more on performance management.
3.2 Simulation and Process Optimization

TeamWorks provides a simulation engine for analyzing process models prior to implementation design and deployment. Simulation mimics the process by stepping through activities in compressed time, keeping track of process metrics while displaying an animated picture of the process flow. The Process Optimizer perspective includes a Simulation Analysis view that not only displays process metrics projected by simulation, but provides recommendations to improve process performance.

Compared to other BPMSs, which support only basic simulation parameters such as participant cost, activity duration, and branching ratios in the flow, TeamWorks allows a richer set of simulation rules. Activity durations not only are specified by mean but by variance and spread...
statistics. Activities can also be assigned user-defined properties such as cost, with both nominal and target values specified (Figure 10).

Because flow connectors (called sequence lines in TeamWorks) can be conditional, frequency of these conditions must be specified for simulation. Similarly, events in the BPD – points where the process waits for a message, timeout, or exception – ignored in most other process simulation models, are included in TeamWorks, specified by a mean frequency, variance, and statistical distribution.

Process Optimizer supports configurable business scenarios combining historical and in-flight performance data. Process models are analyzed in multiple dimensions, including time, count, and path analysis. For example, Figure 11 illustrates time analysis, in which activities are assigned thresholds for metrics like *Average Wait Time*, and colored halos around activities in the diagram report time bottlenecks. Solid red indicates over the specified threshold, and lighter red indicates near the threshold. The *Recommendations* view contains system-generated suggestions for improving performance, such as adding resources to particular process activities.

![Figure 11. Process Optimizer time analysis and recommendations for improvement. Source: Lombardi](image)

### 4. Process Structure and Data

#### 4.1 Process Structure and Terminology

In TeamWorks, the *Business Process Definition (BPD)*, also called the *process model*, specifies the flow and performer of *activities*. BPMN is used as the notation standard for the BPD, and the same model is used for both analytical modeling and executable design. The *performer* of an activity is determined by its swimlane in the diagram. If an activity is in a user swimlane, a
TeamWorks service sends one or more tasks to a user or role. (Multiple tasks will be sent, for example, in a looping activity.) Tasks have envelope properties such as who they are assigned to, their due date, and which users have performed the service in the past. If the activity is in a system swimlane, its implementation, or method, is performed by the TeamWorks engine.

The implementation of a process activity can be either a TeamWorks service, an embedded script (Javascript), or a nested process (i.e., subprocess). Services are modeled in the Service Modeler perspective, and then attached to Activities in the Process Modeler perspective. A business rule is a special type of service (rule service). Most of the implementation properties of an activity pertain specifically to human tasks, and will be described more fully in Section 5.

Services in TeamWorks are frequently implemented as web applications in which multiple web pages or forms, called Coaches, allow the user to perform database lookups or execute other automated functions, and display other Coaches. The logic flow of Coaches and system activities in a service, as well as its event- and exception-handling behavior, is specified graphically in Service Modeler.

Lombardi strongly advocates an iterative design methodology in which the BPD can be “played back” at any time from the design environment, enabling the user to see it work immediately, beginning with crude user interface and functionality, and iteratively refining it. To support that methodology, any activity when originally created can be automatically attached to a default human or system service, based on TeamWorks System data. This allows immediate playback without specifying any implementation detail at all. In this way, TeamWorks erases the boundary that exists between modeling and design in other BPMSs. The Playback feature in the Authoring Environment enables process simulation and testing, which are essential parts of process debugging.

4.2 Flow Objects

The BPMN standard supports three core flow objects, called Activity, Event, and Gateway, and Process Modeler supports all three in BPDs, although not in all the varieties allowed by the full BPMN spec.

4.2.1 Activity Looping

One of the BPMN activity properties supported by TeamWorks is looping, meaning the activity is performed N times, where N may be determined at runtime. Iterations of the loop may be sequential (called simple looping) or parallel (called multi-instance looping). Each multi-instance loop represents a separate task. By default, the running process waits for all instances to complete before it flows to the next step. However, a JavaScript condition may be defined to require only a specified number or fraction to complete, or a time limit.

4.2.2 Events

Unlike many BPMSs that claim BPMN compliance, TeamWorks supports most of the BPMN event types, including message, timer, exception, and terminate, and supports both attached and unattached intermediate events. An attached intermediate event, drawn on the boundary of an activity, signifies that if the event occurs (message received, timeout, exception, etc.) while the activity is running, the activity terminates and flow continues on the path linked to the event. An unattached intermediate event, drawn connected by arrows (called sequence flows in BPMN), signifies that the process stops and waits for the occurrence of the event. See Section 9 for more on event and exception handling.
4.2.3 Gateways and Conditional Flow

Gateways in BPMN are points in the flow where the process instance either follows one of several possible paths, or is split into multiple paths in parallel, as determined by the gateway conditions. TeamWorks supports exclusive branching (called a Decision or Switch gateway), conditional splits, and unconditional splits, and corresponding join gateways. Conditional branches of a split are indicated with a diamond on the tail of the connector (Figure 12). Split and Join gateways are not always paired together; you can use one without the other.

Figure 12. Conditional splits and joins in a process model. Source: Lombardi

4.3 Nested Processes

Nested processes provide a way to link BPDs together in a parent-child relationship. Nested processes are attached to Activities in other Business Process Definitions as part of defining Activity implementation. Nested processes provide a way to encapsulate logically related steps within a process while retaining the high-level view of the parent process. TeamWorks allows nested processes to be called dynamically, i.e. the particular nested process to call is determined at runtime, using a variable.

TeamWorks supports run-time instantiation of sub-processes. When specifying the implementation of a process, the designer can indicate that the process is dynamic and indicate a variable that will contain a list of candidate processes that the user can select at runtime.

Figure 13 - Dynamic Subprocess Invocation Source: Lombardi

4.4 Variables

TeamWorks variables represent data structures or business objects (Figure 14). Most variables are local variables, available only within the process or service level in which they are defined. Calls to activity implementations therefore usually involve data mapping to the called interface. TeamWorks distinguishes between input/output variables, which are used in data mappings, and private variables, which are just used internally to a process or service. For an example of how data mapping works in TeamWorks, see Section 6.
Figure 14. TeamWorks variables represent business objects. Source: Lombardi

Each data element in a variable has an assigned type (string, integer, date, etc.), and TeamWorks provides methods to return values for each type. Thus the year value from a variable `myDate` using the `getFullYear` method is represented in TeamWorks as `tw.local.myDate.getFullYear()`. Variable definitions can specify initialization rules for each element (Figure 15), and each activity in the process can manipulate variable values through Pre and Post assignment rules executed before and after the activity runs.

![Business Process Variables](Image)

Figure 15. Specifying TeamWorks variables. Source: Lombardi

### 4.5 Debugging

TeamWorks provides a graphical debugger from the Authoring Environment using the Process Inspector perspective, based on the same BPMN diagram used for modeling (Figure 16). This fits perfectly with their iterative design methodology, described previously. Runtime information in a TeamWorks process is called execution context. Each process instance has an execution context, which is carried through the run-time process by Tokens. Process designers can view and manipulate individual variable values in the Execution Evaluator view of the Process Inspector perspective.
5. Human Workflow

5.1 Task Assignment

In a BPD, each swimlane stands for a process participant, which in turn represents a logical user group or an external system. Participants are created from the TeamWorks Library, which is the view into the TeamWorks shared model repository. Participants are created at the global level, not at the process level, and therefore can be reused in any TeamWorks process.

At runtime, human tasks show up in the To Do list of assigned users in the Task Manager component of TeamWorks Portal (Figure 17). The rules governing task assignment are specified using the implementation properties of the activity. The task can be assigned either to the next
member of the participant group (in round-robin fashion), the same user who processed the last previous human activity for the instance, or via custom logic using a variable. Multiple users can be assigned the task in parallel by using a multi-instance activity. The priority of a task can be set at design time based on either a predefined level, deadline, or variable, or it can be set dynamically at runtime.

5.2 Coach

The name Lombardi Software was originally taken from legendary football coach Vince Lombardi, and using the task user interface to “coach” untrained users though their tasks remains a distinctive feature of TeamWorks. The Coach Designer (Figure 18) in the Service Modeler perspective provides an intuitive, drag-and-drop editor to build and test those user interfaces, or Coaches.

In TeamWorks, the process of building Coaches is iterative. The first step is building a mockup containing static elements simply to visualize what data is needed and how it should be displayed. Once the basic look and feel is determined, the dynamic interface links graphical elements to real data. At each step, designers can run the Coach by clicking the Playback icon, and test each element. The Preview tab in Coach Designer provides an instant preview of what the runtime Coach looks like in a browser.

Figure 18. Coach Designer allows business analysts to build complex screenflows that guide untrained users in performing process tasks. Source: Lombardi

Typically, a Coach needs to display business data that resides in a data structure, enabling participants to view and interact with the data. Creating tables and binding complex data structures to them is easy in the Coach Designer. Simply drag the variable representing the complex data structure variable from the Outline view and drop it in the Coach. A table is automatically created, and the variable parent element is bound to the table header. The variable properties are each bound to a cell in the table (Figure 19).
Figure 19. Coach Designer creates form tables automatically from TeamWorks variables. Source: Lombardi

A distinctive feature of TeamWorks is the way it models human tasks as a flow of *subservices*, including Coaches, scripts, and integration actions. For example, Figure 20 illustrates a candidate offer activity in a hiring process. It is composed of a Coach that searches a database and selects a candidate, a script to set the selected candidate, and another Coach to complete the offer. Links between these elements are controlled by buttons on the Coaches.

Figure 20. Human tasks modeled as a screenflow in Service Modeler. Source: Lombardi

5.3 Lombardi for Office 2003

In addition to the standard TeamWorks Enterprise offering, Lombardi for Office (Figure 21) provides an alternative framework based on access to tasks and performance monitoring through Microsoft Office rather than a web browser. The product provides integrated task management in Outlook, including a “TeamWorks Today” overview page, searchable process information, a process toolbar for performing typical actions, and automatic synchronization with the server for tasks, folders and searches. Performance monitoring and management is also via Outlook, including display of ScoreBoards and update of process thresholds and policies (see Section 10).

Lombardi for Office supports other Office-specific features, such as SmartTags. When you enter a name, address, company stock symbol, or other identifiable information types in an Office document, SmartTags automatically pops up a menu of options: add the person to Contacts, or retrieve a stock quote. Lombardi for Office adds a menu of process-specific actions from Word or Excel using SmartTags.

Lombardi for Office also supports InfoPath, Microsoft’s XML forms technology. Processes can work with InfoPath forms either online or offline.
6. Integration Framework

6.1 Outbound Integration

Integration with external information systems in TeamWorks is based on a three-level component hierarchy of connectors, integration definitions, and integration points (Figure 22).

Connectors create proxies for specific Java methods, JCA resource adapters, or web service operations. At design time, connectors introspect external information systems and expose the input and output parameters of the selected method or operation.

TeamWorks supplies a basic set of reflection connectors known as System Connectors, providing access to basic file, messaging, and database operations (Figure 23), and developers can create custom connectors from their own Java classes as well.
To use a connector, an instance of it specifying a particular integration operation must first be created in the Authoring Environment. With a reflection connector, selecting the connector Type and specifying the Java class allows the Connector Properties dialog to introspect the available methods for the class and the input/output parameters for each. These are populated automatically by the Reflection API. Similarly, with a SOAP (web service) connector, specifying the endpoint URL and WSDL operation allows the Connector to extract the input/output parameters automatically.

Integration definitions associate TeamWorks variables (or literal values) to each parameter exposed by a connector, creating a reusable integration action in the TeamWorks library.

Integration components (sometimes called integration points) represent instances of integration definitions in a service model of a business process. They map runtime values of process variables to the variables representing integration parameters. Dragging the integration definition
onto the service design canvas creates an integration component. Configuring the component maps TeamWorks variables to integration definition parameters.

For example, in an HR request-to-hire process, the position description (header) is retrieved from a database by an ID field. The database has a web service interface called R2HPositionInfo. In Figure 24, an instance of the web service connector called YourName Get Position Header introspects the web service to select the GetHeader operation and identify its input and output parameters. For each web service parameter, a corresponding TeamWorks parameter is created. In the figure, they are both called positionID.

In the integration definition, TeamWorks parameters are mapped to elements of TeamWorks variables. In Figure 25, positionID is mapped to tw.local.positionInfo.id. TeamWorks automatically creates dialogs that let you test integration definitions by keying in values of these parameters.

![Figure 25. Configuration of an Integration Definition based on the web service connector. Source: Lombardi](image)

Dragging the integration definition into the diagram defining the Generate Offer service (a single activity in the request-to-hire process) creates the Get Position Header subservice (Figure 26), representing the integration point. Values can be mapped to each element of the integration parameters using the Pre/Post tab of the integration point configuration dialog. More complex
mappings of process data may require either a script or XSLT, which would be implemented by other subservices in the Service Modeler diagram.

6.2 Inbound Integration

TeamWorks also supports inbound integration – in which the process responds to external events – through two alternative but similar mechanisms. Undercover Agents (UCA) monitor message events (via HTTP and JMS listeners) and timer events to trigger TeamWorks processes or services. TeamWorks Web Services expose TeamWorks processes or services through SOAP and WSDL.

In a BPD, intermediate message events (either attached or unattached – see Section 4.2.2) can be used for inbound integration. Unattached message events (Figure 27) represent points in the flow where the process waits for a particular message. Each message event must be associated with a TeamWorks Service that is configured to run as an Undercover Agent. Configuration of the Undercover Agent binds it to a message queue and variable (see Section 9.1.1).

Figure 27. Intermediate events used for inbound integration. Source: Lombardi

7. Business Rules

TeamWorks implements business rules through rule services. A rule service is structured as a list of if-then statements, in which each if condition tests a variable against a value (string matching or numerical comparison), and the action associated with the if is a JavaScript expression that typically sets the value of other process variables. For example, if the value of customer is Target and billOverdue is true, then set rebill to true. The first condition in the list to evaluate true triggers its associated action, and the remaining conditions are ignored (Figure 28). Rules services are stored in the TeamWorks Library, enabling users to share and reuse rules across different services and processes.

TeamWorks can also integrate with third-party rules engines such as Fair Isaac Blaze, ILOG JRules, or Corticon, to leverage rulebases that are industry-, domain-, or product-specific, but whose rules are independent of process context. These engines are integrated to TeamWorks via direct API or Web service calls. Lombardi provides packaged integrations to Corticon and Fair Isaac.

Besides offering the ability to write business rules or integrate with external rules, TeamWorks helps business analysts discover patterns in process and data flow that might suggest new rules. The Process Optimizer provides an analysis wizard that discovers potential rules and lets users create those rules automatically. For example, TeamWorks can identify that for an approval activity, the approved status is set to “true” 96% of the time for the largest customer on orders over $10,000. The user can elect to have TeamWorks automatically write a rule that will bypass
the approval activity when the above condition has been met. This simplifies the otherwise complex tasks of identifying new rules that can help further streamline a process.

---

**8. Content, Collaboration, and Case Management**

**8.1 Content Management**

TeamWorks Portal and coaches provide native support for document attachments, and can include links to external ECM applications like Documentum and FileNet. This allows users to access relevant documents and images (e.g., scanned invoice with customer annotations) during process execution. This linkage can also be event-based. As events happen in the underlying ECM systems, TeamWorks can monitor and correlate those events to running processes, or instantiate new processes.

---

**Figure 28. Business Rule editor. Source: Lombardi**

**Figure 29. Collaboration and document attachments in TeamWorks Portal. Source: Lombardi**

© Bruce Silver Associates 2006
8.2 Team Collaboration

The TeamWorks Portal enables team collaboration through a shared workspace. Users can post comments and request help on any task or on the overall process. Documents can be attached to the process at any time and versioned as the process executes. These features ensure the runtime collaboration required to work together on processes that span individuals, teams, and even organizations.

9. Events and Exceptions

9.1 Events Overview

The TeamWorks Event Manager is responsible for all time-based and event-based process activities. Events may be derived from both the process and external applications. Internal events include Timer Events, used to define escalation rules within the process, and Exception Events, which “throw” and “catch” process exceptions. All events, triggers, and deadlines that govern process management can be modeled directly in the process diagram in TeamWorks.

TeamWorks supports Start, Intermediate, and End events as described in the BPMN standard. BPMN events are subdivided according to their trigger type, such as a message, a timer, etc. TeamWorks BPDs support the following trigger types:

- General – no trigger, no input or output parameters; used for manual start of a process or service.
- Message – must be attached to a service linked to an Undercover Agent.
- Timer – based on a time or duration specified by a fixed value or variable
- Exception – “catches” an exception thrown by a TeamWorks Service

The Pre and Post tab enables variable value assignment before or after an Event executes.

![Undercover Agent configuration maps event data to TeamWorks. Source: Lombardi](image)

9.1.1 Message Events

Messages can be received by any running process that contains a Message Event. The actions triggered are specified by attaching to the Event a TeamWorks Service that is configured as an Undercover Agent. A UCA acts as a "wrapper" around whatever functionality it is assigned to execute at run time, described by the TeamWorks Service. For all types of events, the output parameter values from the Service are passed by the UCA to the appropriate business process.
instance. When used as an intermediate event, the message is received by the process instance determined by a variable mapping between the UCA and the Message Event (Figure 30).

### 9.1.2 Timer events

Timer Events can be used to build deadline-based escalation paths in a business process, implement timeout-based exception handling, schedule events and activities that must occur at a particular time, or to pause execution for a certain length of time. Intermediate timer events are typically based on a time interval following the start of the process activity, which can be measured with or without use of a “business calendar” for the activity performer. Timers can also be rescheduled after they have been started, based on other process events or activities.

Figure 31 illustrates an attached timer event used to implement deadline escalation behavior. If the timer event fires, the *HR Approve New Position* activity terminates and the *Escalate Task* activity is enabled.

![Figure 31. Timer events used to model escalation paths. Source: Lombardi](image)

### 9.2 Exceptions

An Exception Event listens for runtime exceptions or throws an exception. An XML representation of the exception that has occurred is made available during the processing of the
post-execution assignments using the `tw.system.step.error` variable. The XML contains a numeric exception code and an exception message in String format. If an exception occurs and there is no Exception Event attached to the Activity where the exception occurs, the exception is propagated up the BPD call stack until it reaches a nested process containing an Activity with an Exception Event attached.

### 9.3 Transaction Management

TeamWorks can support transaction rollback and compensation in two ways. The first is by defining the compensation logic explicitly in TeamWorks. TeamWorks does not support the BPMN Compensate event, but within the process, one can define logic and activities that undo specific process actions, such as sending a follow-up cancellation email or reverting the old value in an external system. Second, TeamWorks can interact with a transaction monitor and commit and rollback transactions as part of the process directly.

### 10. Performance Management

A key strength of TeamWorks is its approach to performance management. TeamWorks emphasizes active business-oriented metrics that are used to optimize running processes in real time and respond to changing conditions. TeamWorks broadly distinguishes tracking from reporting. **Tracking** refers to the logging of selected runtime data values at Tracking Points specified in TeamWorks process models and transferring them to the Performance Server. Tracking also includes automatic generation of `Alert` messages sent to performance management applications called ScoreBoards when triggered by specified runtime conditions. **Reporting** refers to the analytical assembly, formatting, and presentation of tracking data to provide process visibility in ScoreBoards and active optimization through Exposed Process Values.

The TeamWorks architecture features a multiplayer abstraction layer for performance data that allows the physical process model, variables, and components to change and not break reports. Unlike some other BPM offerings, no database work is required to maintain performance management metrics, even when process models change.

#### 10.1 Tracking

Related data elements need for tracking in a business process, such as account or order information, are selected as members of a Tracking Group in the Authoring Environment. Tracking Groups can be envisioned as virtual tables or views in the performance database, specifying fields to track. Tracking groups can be maintained by business analysts or process owners, since no database or process changes are required to add or delete fields.

**Tracking Events** (also called tracking points) are components inserted in process models defining where values of particular Tracking Groups are captured and copied to the Performance Server. A Tracking Event is an instance of a Tracking Group dragged onto the service diagram. The Tracking Event defines both the point where the snapshot of performance data is captured and the mapping from service variable data to named tracking fields. At runtime, the Tracking Event inserts rows into a Tracking Group table on the Performance Server (Figure 32).
Figure 32. Tracking event inserts a row in the tracking group data table. Source: Lombardi

Timing Intervals measure the time elapsed between a pair of Tracking Events, which may be in separate services (but within a single process). Timing Intervals are used to measure process velocity, such as the time it takes for a user to perform a task or a system to update an enterprise information system.

Send Alerts post warnings to a ScoreBoard at particular places in a process. An example might be low inventory for a particular item. Alerts allow ScoreBoard users to take immediate action, and they maintain a record of the exception condition.

Figure 33. Standard Team Performance ScoreBoard provides role-specific interface to performance information. Source: Lombardi
10.2 ScoreBoards

ScoreBoards are the highest level of user interface definitions for reporting. They organize graphical Reports into predefined layouts specific to individual users or roles, providing both functional customization and access control. When users log into the ScoreBoard area, they see all the ScoreBoards associated with their roles (Figure 33). Reports are linked to datasources, and allow users to drill down and filter report data from a ScoreBoard. Reports can also contain Exposed Process Values (EPVs), values of specially declared variables that can be changed on the fly by process owners to optimize process performance. Lombardi provides standard ScoreBoards for Process, Team and Individual Performance out of the box.

The report datasource is specified by selecting an Integration Definition (typically a SQL query on Performance Server data) and an associated Data Transformation to process the returned data (Figure 34). Datasources return raw data as XML, and the transformation controls how the XML is turned into the values matrix required by the chart. Report filters implement a dynamic Where clause for the datasource, allowing a ScoreBoard user to customize the data aggregation at runtime. In addition, report page drilldowns allow ScoreBoard users to navigate from one chart to another, all set up via point-click dialogs.

![Figure 34. Report configuration specifies datasource (left) and chart parameters (right). Source: Lombardi](image)

10.3 Exposed Process Values

Exposed Process Values (EPVs) allow ScoreBoard users to adjust the values of specially declared variables in order to optimize process performance. For example, a manager may want to be able to adjust the purchasing approval levels for a group of customer service representatives. The manager’s ScoreBoard provides access to an EPV called ApprovalAmount that can be adjusted in a range between high and low boundaries. The running process reads the value of ApprovalAmount and blocks all purchases without approval below that amount. The manager can raise the Purchasing Amount Exposed Process Value during a particularly busy purchasing period to avoid being overwhelmed by approval requests.
In the request-to-hire process (Figure 35), the time for salary approval is taking too long, so HR changes the threshold needed for special signoff, using an EPV.

![Figure 35. ScoreBoard users adjust Exposed Process Values on the fly to improve process performance in real time. Source: Lombardi](image)

EPVs play a role similar to rule maintenance applications in Business Rule Management Systems. In practice, multiple values often need to be changed simultaneously to reflect details of the adjusted “business rule.” Beyond just ApprovalAmount, the need for approval might depend on the date of the purchase request, the customer status, and the items requested. A single EPV allows the values of multiple variables used in process logic to be changed simultaneously from a ScoreBoard. EPVs are associated to specific ScoreBoard reports, since the need to adjust them is caused by intelligence received through a report.

In the process logic, EPVs are typically referenced by gateway decision logic and in Javascript. Any changes applied through a ScoreBoard take effect immediately without versioning and redeploying the process.

11. Industry Solutions and Services

TeamWorks offers Process Templates intended to accelerate deploying BPM solutions and provide insight into implementations of diverse business processes. Each Process Template provides an executable process that includes BPMN process diagrams, roles, activities, rules, user interfaces, key metrics, simulation information, and best practice dashboards. The templates are easy to extend and customize to meet specific customer needs. A current listing of TeamWorks Process Templates includes:

**Accounts Payable/Receivable**
- Credit Card Dispute Resolution
- Invoice Approval Processing
- Wire Transfer Mgmt
- Procure to Pay
- Deductions Management
- Billing Exception Resolution

**Procurement/Supplier Management**
- Procure to Pay
- RFQ Management
• Deductions Management
• Inventory Management

Sales/Marketing
• Incentive Compensation Management
• Price Management
• Territory Management
• Collateral Development

Finance/Compliance
• Quarterly Financials Reporting
• Operating Plan Development
• Asset Transfers
• SOX – Computer Access

Human Resources
• Employee Self Service
• Manager Self-Service
• Employee Evaluations

Operations (Vertical)
• Underwriting Exception Handling (Financial Services)
• Capital Project Management (Financial Services)
• Policy Administration (Insurance)
• First Notice of Loss (Insurance)
• Plan Configuration (Insurance)
• Case Management (Life Sciences)
• Chargeback Reconciliation (Life Sciences)
• Grant Management (Life Sciences)
• Sample Management (Life Sciences)
• Order Management (Communications/High Tech)
• Service Delivery (IT)
• Trouble Ticket Management (IT)

In addition, Lombardi ISV and System Integrator partners offer a diverse set of vertical process templates and applications:

• Fiserv Insurance Solutions has developed a set of templates for key insurance processes, including First Notice of Loss and Underwriting;

• Cognos is embedding Lombardi’s TeamWorks BPM software in its analytic applications. The initial result will be Cognos Workforce Performance, the first of Cognos’ reengineered suite of analytic applications;

• Paisley Consulting is developing their next generation of Compliance applications with TeamWorks as the core process engine.

In addition to traditional planning and implementation services, Lombardi offers the Lombardi On-Demand Assistance (LODA) program. At the core of the LODA program is a dedicated call-center that provides assistance with solution and process-specific questions and problems. In addition, LODA provides a package of education and periodic on-site visits to ensure that customers are successful deploying BPM.
In keeping with their focus on iterative development, Lombardi education and certification offerings include not only skills training but also project planning and management education. The Implementation Framework class teaches project managers how to organize iterative development cycles and how to use Six Sigma tools like Failure Modes and Effects Analysis (FMEA), Cause and Effect Matrices. Lombardi believes that Six Sigma disciplines can be used in conjunction with BPM to better prioritize, scope and execute projects. Their services team includes Master Black Belt, Black and Green belts.
12. Analysis

12.1 Overall Assessment

Lombardi TeamWorks distinguishes itself from the BPMS pack by emphasizing business-oriented “visibility” at every stage of the process lifecycle, from analysis to design/debug to performance monitoring in execution. The visual metaphor for all aspects of process modeling and management is the BPMN diagram, which is used for modeling and simulation analysis, process design, BAM design, test and debug, instance tracking, and performance optimization. Process analysts, designers, task participants, and process owners all share a common visual BPMN representation of the business process.

Unlike other BPMSs that support a simple subset of BPMN shapes and semantics, TeamWorks supports essentially the full BPMN semantics, including intermediate events. While the ability for processes to listen for and respond to external events is common in BPEL- or BPML-based engines, TeamWorks today stands alone among BPMSs oriented toward human workflow in its support for BPMN events. Also, because BPMN makes event-triggered flows and exception handling visible in the process diagram, TeamWorks makes it all visible to analysts and developers alike.

A second distinguishing characteristic is support for rapid iterative development. Other vendors frequently pay lip service to this implementation style, but Lombardi supports it concretely with the ability to instantly “play back” activities and process fragments even early in the modeling phase, creating default components as necessary that will be refined later on. Lombardi emphasizes a project methodology in which a simple version or fragment of the process is piloted very quickly, with additional features and richness layered on iteratively after that.

A third emphasis is the user experience for task participants. In TeamWorks, task user interfaces are not simple forms. Instead, they are screenflows composed graphically using forms (“coaches”) and data manipulation using scripts and integration components. TeamWorks gives non-technical users the tools to create and maintain these task interfaces without programming, and play them back at any stage of the development cycle.

Fourth, TeamWorks stands out in its focus on user-defined performance monitoring and real-time optimization. Unlike most BPMSs, in which defining custom business metrics requires detailed implementation knowledge and database expertise, TeamWorks supports specification of tracked performance data in the high-level process model, before the implementation of process activities is even defined. Moreover, the TeamWorks Process Optimizer offers a unique combination of process simulation, in-flight monitoring, and historical analysis to not only visualize bottlenecks but eliminate them based on system-generated recommendations such as substitution of a business rule, or reconfiguring task resources. Based on Process Optimizer and performance management dashboards (ScoreBoards), TeamWorks also allows process owners to reconfigure the process on-the-fly using exposed process values, variables whose values can be manipulated in-flight through a web interface.

Finally, TeamWorks for Office new extends BPM to the casual user desktop via Microsoft Outlook and Word, supporting offline work, InfoPath forms, and other features of the Microsoft Office environment.

Together, these capabilities make TeamWorks an excellent choice for human-centric processes, particularly where rapid implementation, easy maintenance by non-technical users, and performance visibility and optimization are important.
12.2 Use Case Fit

12.2.1 Basic Workflow

Fit: ● ● ● ●

Checklist:

- **Rich forms user interface.** A key strength of TeamWorks.
- **Process design and maintenance by non-programmers.** Yes. TeamWorks distinguishes process modeling, the sequence of activities in the BPMN diagram, from service modeling, defining the implementation of those activities. All of process modeling and most of service modeling, can be performed by non-technical users.
- **Easy to deploy and maintain.** Yes, this is a major emphasis of the design environment.

12.2.2 Content Lifecycle

Fit: ● ● ● ▼

Checklist:

- **Support for document attachments and viewers.** Yes.
- **Support for content management library services: check-in/check-out, versioning, metadata search.** Not directly, but integrates with leading ECM repositories like Documentum and FileNet.
- **Support for scalable ECM repositories.** See above.

12.2.3 Complex Collaborative

Fit: ● ● ● ▼

Checklist:

- **Support for collaborative document review and discussion.** TeamWorks Portal supports a shared workspace where users can share comments or request help on a process task.
- **Support for unstructured/ad hoc flow and offline work.** Some. Chained processes can be spawned at runtime and correlated with the original process for performance tracking. Lombardi for Office supports offline work via Microsoft Outlook.
- **Integration of online team rooms.** No.
12.2.4 Case Management

Fit: ● ● ● ●

Checklist:

- **Electronic case folder of independent work objects.** Can be implemented but not a native construct.
- **Ability to add case objects and flows at runtime.** Some support for this via dynamic subprocesses and events.
- **Content management integration.** Not native, but integrates to third party repositories.

12.2.5 Production Workflow

Fit: ● ● ● ●

Checklist:

- **Support for shared queues and rule-driven task assignment.** Yes, plus task priority attribute.
- **Performance optimization through simulation, analytics, and real-time escalation.** The integration of these capabilities is a distinguishing strength of TeamWorks.
- **Business rules.** Yes, native rule engine or packaged integration to Fair Isaac or Corticon.
- **High-performance document retrieval/parsing.** No.

12.2.6 Transactional/STP

Fit: ● ● ● ●

Checklist:

- **Rich integration infrastructure, including adapters and enterprise service bus.** Limited. Going beyond built-in system adapters may require Java development, although TeamWorks supports iWay third party adapters. No ESB.
- **Complex business objects, data transformation, and business rules.** Variables represent business objects, but mapping requires scripts or user-built XSLT. Yes for business rules.
- **Comprehensive event management and automated exception-handling.** Support for event-triggered flow and exception handlers through BPMN intermediate events.
- **Industry solution templates with prebuilt objects, transformations, adapters, protocols, performance metrics and reports corresponding to industry standards and best practices.** Yes; mostly human-centric processes.
12.3 Process Lifecycle Fit

Lombardi TeamWorks is a good fit in organizations where BPM is viewed as a business-driven and business-maintained responsibility. Lombardi’s methodology features business-driven modeling of both process flows and performance metrics, with iterative design and deployment cycles enabling user feedback throughout the design phase. There is no distinct “handoff” between business and IT; the line between modeling and implementation design is thin and blurry. A common design tool and notation is used by business and IT, and leveraged for instance tracking and performance optimization as well.

Moreover, business rules and exposed process values give process owners the opportunity to tweak processes after deployment, even instances in-flight, to maintain peak performance.

About the Author

Dr. Bruce Silver is an industry analyst and consultant specializing in BPM and content management technology. He has advised both users and vendors of BPMS software since 1994 as an independent analyst, and prior to that as VP at the analyst firm BIS Strategic Decisions (which became Giga Information Group, now part of Forrester Research). He writes the BPMS Watch column on BPMInstitute.org and the Change Agent column for Intelligent Enterprise magazine. He is also the BPMS Track chair at the Brainstorm BPM/SOA Conference series, and provides training on BPM technology through the BPM Institute. In addition to the 2006 BPMS Report series, Bruce Silver publishes numerous free reports on BPM and content management technology available through his BPMS Watch blog, www.brsilver.com/wordpress.

Comments on this report and requests for future reports are welcome. Contact bruce@brsilver.com.