INTRODUCING REAL-TIME BUSINESS CASE DATABASE
An Approach to Improve System Maintenance of Complex Application Landscapes

Oliver Daute
SAP Deutschland AG & Co. KG, Germany

Keywords: RT-BCDB, Business Case, Business Process, ITIL.

Abstract: While system maintenance of single systems is under control nowadays, new challenges come up due to the use of linked up software applications in order to implement business scenarios. Numerous business processes exchange data across complex application landscapes, for that they use various applications and compute data. The technology underneath has to provide a stable environment maintaining diverse software, databases and operating system components. The challenge is to keep the application environment under control at any given time. The goal is to avoid incidents to business processes and to sustain the application landscape with regard to smaller and larger changes. For system maintenance of complex environments information about process run-states is indispensable, for example when parts of a system environment must be restored. This paper introduces the *Real-Time Business Case Database* (RT-BCDB) to control business processes and improve maintenance activities in complex application landscapes. It is about a concept, to gain more transparency and visibility of business processes activities. RT-BCDB stores information about business cases and theirs run-states continuously. Service frameworks such as IT Service Management (ITIL) can benefit of RT-BCDB as well.

1 INTRODUCTION

More transparency inside application landscapes is required since concepts like Client/Server architectures or service-oriented architecture or IT service management enable applications to compute and exchange data with less respect to the information technology underneath. RT-BCDB is an approach, to collect and provide information about business processes in heterogeneous application landscapes. In RT-BCDB all information of run-states of active business processes is collected and stored synchronously. This information supports the maintenance activities and assists the administration, especially after a system failure. System failures require detailed investigations about the impact on business activities before a system restore can take place. This means processes which were disrupted must be identified and must be included in the recovery process. Therefore having detailed information is a prerequisite for the analysis and for restoring back to a consistent state.

From the perspective of a business case a consistent state requires more than data integrity on database level. Also dependent interfaces or single process steps must be included in considerations. Those can halt in an inconsistent state anywhere in an application environment. The challenge is, to provide knowledge about business processes and their dependencies to support the system administration team in their work. Yet there is no open concept available.

The use of information about business processes in the application environment is essential for planning, changing and optimizing the landscape.

This paper presents RT-BCDB, an approach to support the administration of application landscapes, and delivers information for the IT change-management process. RT-BCDB is a source for monitoring tools and for business processes as well. Knowledge about run-states of business processes is important information for maintenance and for control of business processes. In the next chapter, some terms will be explained followed by an introduction to the idea of RT-BCDB.

2 TERMS

The term, *application landscape,* encapsulates a collaboration environment of hardware and software technologies with a common purpose to provide an infrastructure for business processes for an enterprise. In relation to hardware and operating software components, an application landscape can consist of ERP software, various legacy systems, data warehouses, as well as middleware for exchanging data and connecting software applications. Other expressions for application landscapes are *application infrastructure* or *application environment*. Whenever one of those terms is used it refers to the same hardware and software construction with a different point of view.

A *business case* describes a procedure of activities to fulfil enterprise tasks. A business case can consist of several processes which can be performed on different systems. A process itself is composed of process activities. Again each activity can be implemented as well by sub activities. Another word for process activity is process step.

A *business process* consumes data or calculates them and can invoke IT services. The focus lies on the application level, with respect to the enterprise needs. Often business processes such as invoicing, are called *core business process*.

Another more technical term is *system maintenance activity*. It relates to ongoing tasks on the system administration level. For instance maintenance activities are system updates, customer support, problem analysis and planning and can be distinguished in reactive or active activities. RT-BCDB supports active as well as reactive maintenance activities.

An *enterprise solution* is built by several software components and is outlined by the business needs and requirements. The business processes are reflected within an enterprise solution.

3 THE IDEA

The concept of “Real-Time Business Case Database” (RT-BCDB) introduces a source of information about business processes, which run within an application landscape. The information stored in the database is essential, for the IT system administration as well for the application support (business users, business process reliability). In RT-BCDB each business process is described as a business case. At run-time the business processes write their run-status into the database. Due to this mechanism RT-BCDB contains the real-time run-states of all active business processes. That leads to more transparency about the activities in the application environment and offers additional options for optimization to increase the quality of maintenance activities.
The picture depicts the complexity of an application environment consisting of several layers. The hardware layer consists of servers, storage and network components. Application programs, database systems, and tools to exchange data belong to the software layer. On top of that, the business process layer combines applications and data sources to an enterprise solution. The enterprise solution is the application environment to fulfil the business requirements.

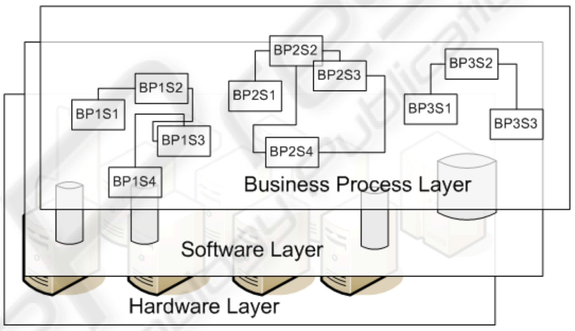

Figure 1: Layers of an enterprise solution.

As shown, a single business process is able to invoke process activities across the whole application landscape, on more than one server and uses different applications and exchanges data to achieve its goals. In a 1-server and application environment it is much easier to keep track of the run-states of the active processes. But how to observe business processes in a complex application environment? If for instance a server fails, or a database stops processing, then several business processes can be affected. In such a situation it is quite difficult to determine the impact to the process activities of the enterprise solution.
RT-BCDB is an open concept for all application environments to answer the question which business processes were interrupted and how to restart them. RT-BCDB provides reliable information about the run-states of processes, their process dependencies and provides useful information for maintenance activities. Some software manufactures already offer proprietary solutions for their business applications and try to make single process activities visible. But most tools can not be used in heterogeneous application landscapes. Primarily, these tools are used to monitor activities of their software

applications. An overall and open concept like RT-BCDB will help to better understand the dependencies and the state of the application environment.
The need to track changes is not new. It is an essential task of database management systems to log transaction activities. Transactions like updates, deletes or inserts change the state of a database with every execution. These changes are recorded in a kind of protocol, the log file. Database logs keep track of the changes, and speed up processing. While logs are written synchronously, the databases can continue their work by using cached data. Because all changes to data are logged instantly, the mechanism guarantees the data consistency. If necessary to avoid inconsistency on data level, databases can be recovered to a consistent state, by identifying unfinished transactions. Also transaction logs can be used to restore a database. With a database backup and a right sequence of logs it is possible to restore a database to a specific point in time, for instance before a table was deleted by mistake.

Keep that database mechanisms in mind, and think of a complex application landscapes, with various business process activities. How would you restore just parts of the application landscape with regard to disrupted business activities? A knowledge database like RT-BCDB with detailed information about run-states could provide the answer to this question.

The idea of tracking activities is adapted to RT-BCDB, the real-time business case database, to support the system maintenance and business level.

RT-BCDB is able to support following IT issues: system updates, recovery, monitoring, transparency, reporting, planning and optimization the time schedule of business processes.

4 ARCHITECTURE OF RT-BCDB

To put it simply, RT-BCDB consists of a directory of named business cases and a run-state table containing run-state information.

The directory lists known business cases and delivers information about general usage, processes and process activities as well as a history of the run-times. As described a business case can invoke several processes across the whole application landscape. Therefore it is important to know on which systems the processing took place. Especially for recovery it is indispensable, to understand the impact to business cases.

Another part of the RT-BSCD architecture is the run-state agent. Software-applications send run-state information to an agent, or the agent itself can evaluate run-state, and writes them into the run-state table.

Due to this mechanism, the run-state table reflects always the status of processes within the application environment, and shows the progress.

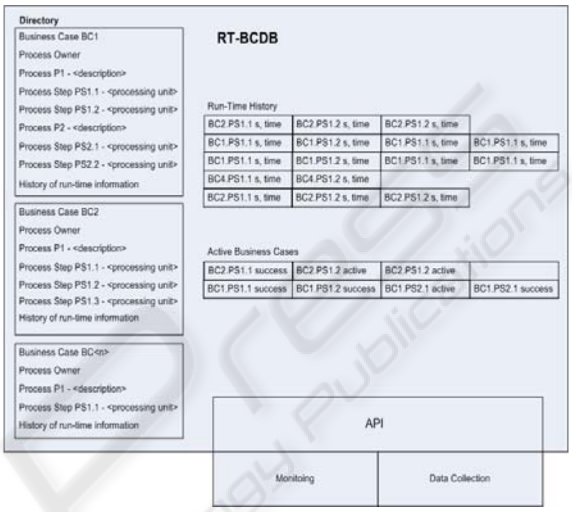

Figure 2: Architecture of RT-BCDB.

Figure 2 outlines the architecture of the business case database. The directory names business cases and gives information about process owner, process and processing units. Each business case has its own unique identifier. Whenever a business case is active it can be checked against the directory. Information can be added and inconsistencies can be discovered.

A business case consists of one or more processes which can contain additional activities, called process step or sub-activity. The run-time history stores executions of business processes. Each row of the table belongs to a terminated business case. An entry describes the business case, its process activity, the state of termination and the run-time. For instance `BC2.PS1.1 s,time` stands for the process step 1 of process 1 of business case 2 terminated with state `s` (successfully), `time` in milliseconds.
The table of active business cases contains run-states of business cases running currently in the application environment.

The abbreviation `BC1.PS2.1 active` stands for process step 1 of process 2, and business case 1 is still active. Additional states are for instance defined as `success`, `stopped`, `abort` or `wait`. More states are imaginable regarding the need of business cases in a specific environment.

The API is the interface for data collection and monitoring. The agent collects information of run-states, and writes them with additional information into the database table of active business cases. How a run-state can be determined will be described in the next chapter. The monitoring interface is used to connect third party applications. These applications can represent the data in form of available business cases or can monitor the current progress of a business case.

5 COLLECTING RUN-STATES

Several options are given to collect run-state information of business processes. The layers of the application landscape provide different views of process activities. On the hardware and operating system layer, information about progress of a process can typically be found in log files. Usually applications write processing messages into flat log files. Messages can reflect the current state or are usable to determine a run-state. An agent for example can search for a specific pattern like `start` or `end` in a log files can be mapped to a corresponding status like `active` or `terminated`. On the operating system level, applications fork processes which can be monitored as well. A start or the termination of an operating system process can point to a run-state. Both options are quite simple methods usable in any application landscape environment. The implementation is easy to realize and comes without the need to adapt applications. It can be used to get a general overview of process activities and to collect information about the progress. The disadvantage is, that no detailed information about process steps is collectable with this method. The advantage is that less effort is required to setup the monitoring. And for smaller environments those methods are sufficient enough to monitor process activities.

The next level to collect run-state information is the application layer. It is much closer to the business cases as the previous one. Most applications and databases have their own monitoring tools to measure and make activities visible. Monitoring tools are used to check for intensive data load, monitor processes or to prove the parallelization of user work and can therefore be used to collect run-state information. Data produced by the monitoring tools can often be extracted to XML or flat files. The extract is useful to identify runs-states of process activities on process level. Not each and every detail of process activities can be collected here but it completes the information collected on the hardware level and is easy to obtain as well. Collecting information on this layer can be done without further financial investments. No development work is required. Only little work needs to be done, to evaluate the run-state out of the extracts.

For sure the best level to determine the run-state of processes is the business process layer itself. Whenever a business process starts, stops or waits, the application should send a message with its state to RT-BCDB. Due to this detailed information of process-activities, processes can be made visible. To insert run-state messages, two options are available. The best method is to insert RunControl commands into the application. This requires modification to the source code.

The run control commands send the business case ID, run-state and the process step to the agent.

```
RunControl(process step,run-state)
```

Another option would be inserting run control commands, is to start and stop applications out of a batch processing mechanism or a script file. Within these scripts run control commands can be set. For instance the start and the end of an application is notifyable. But detailed information within the application run is difficult to derive. The advantage, no modification to the application is required.

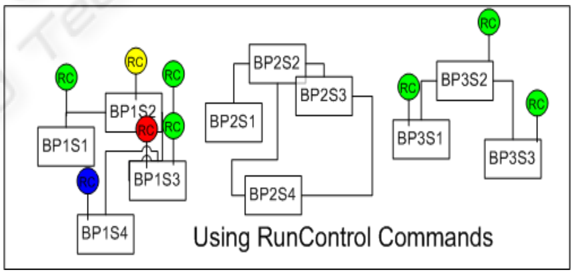

Figure 3: Using RunControl Commands.

As described, several options are available to collect run-states of process activities. Each source of process activity must be evaluated first. After that run-state information and must be sent to the RT-BCDB.

Therefore agents are an important part of the concept. They are installed on each system and belong to the application landscape environment. The tasks of the agents are dependent on the kind of source of information. On hardware and software layer the agents have to evaluate XML, plain log files or extracts of the application tools. The agent inspects the sources and tries to identify run-state using pattern matching. Whenever possible, they map a hit to a corresponding run-state for business process.

On the business process layer the agent receives the run-states from the call of the run control commands. All run-state values must be sent immediately to the RT-BSDB where they are being stored.

6 IMPROVING SYSTEM MAINTENANCE

Subject of this article is RT-BCDB which will improve system maintenance in complex application landscapes significantly. The run-time and run-state information stored in RT-BCDB provides important knowledge about process activities and dependencies within the application infrastructure. This knowledge base helps the system administrators to monitor and optimize the schedule of business cases. From the point of the hardware layer, the system activities can be made visible by querying the RT-BCDB. High and low load in the application landscape are detectable and action can be taken. With that knowledge about distributed process activities, for example, a rebalancing of application load is easier to perform. In case of performance bottlenecks moving a business case to run at a different time with lower overall activities can reduce the overall performance of the application landscape significantly. This is just one advantage of RT-BCDB. It makes reasons for application load on systems identifiable. This is an example of active system-administration on how RT-BCDB is able to improve system maintenance.

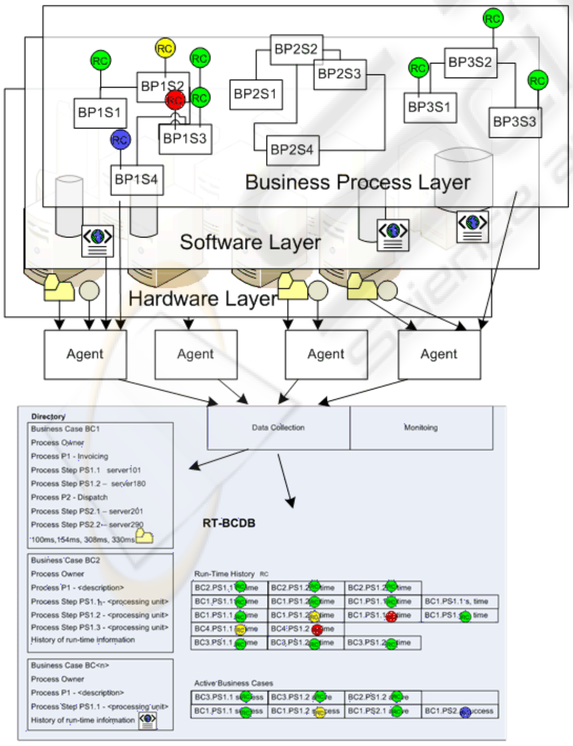
RC
BP1S1
BP1S2
BP1S3
BP1S4
BP2S1
BP2S2
BP2S3
BP2S4
BP3S1
BP3S2
BP3S3
Business Process Layer
Software Layer
Hardware Layer
Agent
Agent
Agent
Agent
Data Collection
Monitoing
Directory
Business Case BC1
Process Owner
Process P1 - Invoicing
Process Step PS1.1 server101
Process Step PS1.2 – server180
Process P2 - Dispatch
Process Step PS2.1 – server201
Process Step PS2.2 – server290
100ms,154ms, 308ms, 330ms
Business Case BC2
Process Owner
Process P1 - <description>
Process Step PS1.1 - <processing unit>
Process Step PS1.2 - <processing unit>
Process Step PS1.3 - <processing unit>
History of run-time information
Business Case BC<n>
Process Owner
Process P1 - <description>
Process Step PS1.1 - <processing unit>
History of run-time information
RT-BCDB
Run-Time History
Active Business Cases

Figure 4: Collecting run-states.

Maintenance tasks like updates and upgrades of the application landscapes also require detailed information about the involved processing units of business processes. In RT-BCDB all information is about processing steps and their processing units are listed. Before a change of a processing unit can take place, all business cases using this unit have to be inspected to determine technical prerequisites for the processes are still valid after a change. If the requirements are met an update is feasible.

The business case database improves transparency and makes monitoring issue of business cases easier to gain knowledge for optimization.

Now we get to the usage of RT-BCBD in case of failure situations. Reactive system maintenance is required whenever an incident in the application landscape occurs. Incidents are for instance an application stops processing, performance problems or an error on hardware operation level occurs. In case a hardware problem occurs, a system has to be replaced or a database needs to be recovered.

Let's assume following situation. Several business cases run within an enterprise environment, when suddenly a database stops processing, see picture.

At first, the last run-state before the crash must be evaluated to identify business cases, which were active at time of failure. A closer look to the directory of business cases of RT-BCDB shows that the processing unit on which the database runs is used by two business cases BC1 and BC2. Now, the run-state table gives more information about the progress of the involved business cases at time of failure. For both business cases failures were reported. They need to be restarted after the database problem has been resolved. For BC2, no further action is required since it is the last processing step of the business case. The process step of BC1 has a successor that needs to be monitored after restart. This is a short example how RT-BCDB supports a detailed analysis of a failure situation. Without this information it would be much more difficult.

7 APPROACH TO RT-BCDB

The real-time database is closed concept usable in almost any application landscape. The question, how to make an application landscape or software-application compliant to RT-BCDB will be answered next.

Without any modification to application source code, information can be retrieved on the hardware and software layer in logs and in extracts of application tools. This is the first step to introduce a RT-BCDB. Additional work is needed to enrich the application with run control commands. This is surely the most valuable step. Therefore application should be developed with regard to run-state information for RT-BCDB. The idea is to place run control commands at each line in the source code, whenever a process activities started, stopped, waited or failed.

It's irrelevant, which process activities will later be part of a business case. This will be determined later by business process designers who are able to define the relevant process-activities which need to be monitored. The sequence of processes steps has to be added into the RT-BCDB business case directory. Run control commands of processes which are not used for business cases can be stored in a separate run-state table for debugging purposes.

8 ITIL

ITIL is a library for IT service management. In ITIL a representation of systems, services and relationships in stored in the configuration management database, CMDB. The CMDB is part of the configuration management and describes an application landscape in form of a virtual image. Management processes like incident and problem and change management make use of the virtual image, to plan changes and improvements.

The concept of RT-BCDB proposes a more detailed view of run-state information of business processes and is therefore a reasonable extension for CMDB. Necessary structures to store information are already described by ITIL using configuration items (CI). One proposal is to extend the description of CI's to store run-state data for RT-BCDB.

9 CONCLUSIONS

RT-BCDB is a concept for complex application environments to gain more transparency of business process activities. It provides adequate information for system maintenance especially in problem situations, and for optimization and planning purposes.

For sure a lot of work needs to be done to implement such a concept, especially if software applications need to be adapted to send run control commands. But, with the constant increasing complexity of application landscapes, a mechanism as described is indispensable to keep an application infrastructure maintainable.

REFERENCES

ITIL, IT Infrastructure Library, http://www.itsmf.net IT Service Management

Kobbacy, Khairy A.H.; Murthy, D.N. Prabhakar, 2008 *Complex System Maintenance Handbook,* Springer Series in Reliability Engineering

Scheer, 2003. *Modellierungsmethoden Metamodelle Anwendungen*. Springer.

Daute, O., 2004*: Representation of Business Information Flow with an Extension for UML*, ICEIS 6th Conference on Enterprise Information Systems