D5.1

Version:	0.7
Date:	2008-07-30
Author:	UNITN
Dissemination status:	PU
Document reference:	D5.1

State of art in the field of Adaptive Service Composition Monitoring and Management

Project acronym:	COMPAS	
Project name:	Compliance-driven Models, Languages, and Architectures for Services	
Call and Contract:	FP7-ICT-2007-1	
Grant agreement no.:	215175	
Project Duration:	01.02.2008 – 28.02.2011 (36 months)	
Co-ordinator:	TUV	Vienna University of Technology (AT)
Partners:	CWI	Stichting Centrum voor Wiskunde en Informatica (NL)
	UCBL	Université Claude Bernard Lyon 1 (FR)
	USTUTT	Universitaet Stuttgart (DE)
	TILBURG UNIVERSITY	Stichting Katholieke Universiteit Brabant (NL)
	UNITN	Universita degli Studi di Trento (IT)
	TARC-PL	Apera sp. z o.o. (PL)
	THALES	Thales Services SAS (FR)
	PWC	Pricewaterhousecoopers Accountants N.V. (NL)

This project is supported by funding from the Information Society Technologies Programme under the 7th Research Framework Programme of the European Union.

Project no. 215175

COMPAS

Compliance-driven Models, Languages, and Architectures for Services

Specific Targeted Research Project

Information Society Technologies

D5.1 State of art in the field of Adaptive Service Composition Monitoring and Management

Due date of deliverable: 2008-07-31

Actual submission date: 2008-07-30

Start date of project: 2008-02-01 Duration: 36 months

Organisation name of lead partner for this deliverable: UNITN

University of Trento, Italy

Revision 0.7

Project funded by the European Commission within the Seventh Framework Programme		
Dissemination Level		
PU	Public	X
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

History chart

Issue	Date	Changed page(s)	Cause of change	Implemented by
0.1	2008-05-30	All sections	New document	UNITN
0.2	2008-06-20	All sections	PART 1	UNITN
0.3	2008-07-11	All sections	PART 2 + Revision based on reviews	UNITN
0.4	2008-07-17	All sections	PREP + Revision based on reviews	UNITN
0.5	2008-07-20	Section 4.2	Research approaches addition	UCBL
0.6	2008-07-24	Section 4.2 and Conclusion	Revision	UCBL
0.7	2008-07-30	Section4 and References	Revision	UCBL, UNITN

Authorisation

No.	Action	Company/Name	Date
1	Prepared	UNITN, UCBL	2008-07-30
2	Approved	TUV	2008-07-31
3	Released	TUV	2008-07-31

Disclaimer: The information in this document is subject to change without notice. Company or product names mentioned in this document may be trademarks or registered trademarks of their respective companies.

All rights reserved.

This document reflects only the authors' view. The European Community is not liable for any use that may be made of the information contained herein.

Contents

List of figures

List of tables

Abstract

Monitoring service executions and service composition executions (i.e. business processes) is the first step toward understanding the real behavior of services and service compositions and, hence, toward comparing a detected behavior with the desired one. Comparing real execution data with desired (compliant) behaviors, in turn, enables one to assess whether services and service compositions, are being executed according to agreed upon rules and policies or whether there are discrepancies. That is, business execution monitoring is the cornerstone for compliance assessment.

This deliverable summarizes the state of the art in adaptive service composition monitoring and management in a comprehensive overview spanning commercial monitoring and management products and academic research approaches. We first introduce a compliance management life cycle, as seen from WP5, in order to position the deliverable with respect to other works. We then overview commercial products and research approaches that allow the monitoring and management of services and service compositions. For this purpose, for both products and research approaches we first introduce a set of dimensions, which we then use to characterize the products and approaches and which allow us to understand commonalities and differences. We then analyze some of the most prominent business intelligence and reporting suites, which can be used in the development of the governance dashboard for the visualization of monitoring results. Next, we identify best practices, discuss the lessons learned, and identify which solutions are suitable for the work on compliance governance in WP5.

1. Introduction

The last years have been characterized by increasing investments by companies and Public Administrations in so called Business Activity Monitoring (BAM) solutions, a term coined by Gartner in 2002 [McCoy02]. The aim is that of providing decision makers with company-wide, real-time and historical business information, e.g. in form of business performance indicators, in order to improve the speed and effectiveness of business operations and, eventually, enable users to make better informed decisions. The main goal of business monitoring solutions is, hence, to enable decision makers (e.g. managers or heads of departments) to view how their company or administration is performing, which business activities are generating value or revenue, which do not, and which problems or risks the company is exposed to.

As BAM and similar monitoring solutions matured, compliance management emerged as a natural extension of such kind of automated support to decision-making or controlling. Indeed, while the visibility of precise performance indicators and the tracking of business objectives may enable decision makers to assess not only whether business is performing well, but also whether business is performing according to laws, regulations, standards, or internal regulations, this assessment is in most cases still a manual task. In addition, this task can typically only be performed by experts (e.g. by lawyers or specialized consultants). This applies to the case where a company uses (automated) information systems that allow the collection of suitable performance indicators and of runtime data, necessary to assess the level of compliance with such regulations. In most cases, however, such systems are not readily available, and – if any – compliance controls are performed manually, typically by checking only part of the running business (e.g. 5% of a clinic's patient records) in order to guarantee a statistical confidence level of let's say 95% of compliance.

But what are the reasons for non-compliance in business execution? We identify three main reasons: First, a company might simply not be fully *aware* of which regulations and laws apply to its specific business, therefore neglecting important controls or not providing the right level of accountability. This typically exposes the company to the risk of fines or of low performance. The recent burst of Enron in the United States or the burst of Parmalat in Italy, however, have raised the awareness for compliance in business execution and, also, laws have been changed accordingly. Second, a company might be aware of all the necessary regulations and laws to be taken into account, but it still might fail in *implemeningt* its business processes in a compliant way. It might indeed be hard to understand which concerns are pertaining to a specific business and how to implement the respective measures in the own running business. Third, a company might have implemented all its business processes (this encompasses also manually executed business processes, not necessarily automatically supported business executions) fully compliantly; however, business is typically executed by human and automated actors, and both may *fail* in fully respecting the predefined process. In order to be able to detect such failures and, subsequently, to enable reparation of violations to maintain compliance, it is important to constantly monitor the running business, so as to enable either a human or an automated actor to compare the real execution of the business with the compliantly designed business practice.

The monitoring of the running business therefore plays a crucial role in compliance management, but typically it is hard (if not unfeasible) to monitor whatever activity in a company's everyday business. Specifically, if parts of a business are performed manually, e.g. a secretary registering new clients in a cartulary and making photocopies of the new customer's documents, the respective activities cannot be automatically validated, unless the secretary explicitly enters each completed task into a dedicated application (but still the application could not be sure that the declared activities have really been performed). In this deliverable, we therefore focus on the monitoring of those activities in a company that can effectively be monitored by a software application. More precisely, the IT focus of this deliverable is on business that is executed in a service-oriented architecture (SOA).

Business processes and activities in SOAs are typically implemented as service compositions or web services executed in a distributed fashion. Parts of the compositions or services used in the implementation of the business may therefore also be outsourced and, hence, operated by external partners. Such distributed environments pose new challenges to the monitoring activity and to compliance management in general, as there is no single source of control for all the running processes and services, which aggregates all the execution data of the company.

In order to understand which are the important challenges and problems and which are the current solutions or best practices in service composition monitoring, in this deliverable we focus on how existing industrial products approach the problem and on what kind of instruments and features they provide to their users. For a better understanding of actual solutions and approaches, we then focus on relevant research approaches.

1.1. Purpose and scope

The purpose of this deliverable is to understand the *best practices* in service and adaptive service composition monitoring, in order to apply the lessons learned in the development of the WP5 monitoring infrastructure and, possibly, identify existing, reusable works. It is however clear that, due to the innovative nature of the research proposed in WP5, it will not be possible to simply apply off-the-shelf commercial solutions without proper adaptation to the particular context of the WP5. Nevertheless, understanding current practices, technologies, and solutions is essential for identifying those solutions that fit best the requirements posed to WP5.

This deliverable focuses on the monitoring of service and service composition executions. That is, we are interested in understanding how *runtime information* about services and compositions can be *collected*, *formalized*, and *stored* in order to understand their runtime behavior. Collected data can then be used to *adapt* the running business in order to guarantee compliance, e.g. by enforcing desired behaviors or compensating for undesired behaviors. In WP5, we specifically focus on how to present collected data to the *human user*, so that the user is aware of possible compliance problems and, hence, able to mitigate possible risks (e.g. by enforcing new activities or re-designing business practices), if necessary. WP5 does not focus on the *automated support* for adapting a running business, i.e., we do not focus on the automated enforcement of compliance. It is however worth to note that individual outputs computed in WP5 (e.g. KPIs) could be used by other WPs to support automatic enforcement.

Relationship of D5.1 with D2.1: Deliverable D2.1 [D2.1], which is being developed in parallel with this deliverable, also contains a discussion of monitoring and related problems in COMPAS. It is important to note that, although this might apparently result into overlapping contents across the two deliverables, the intent of the two investigations is different. D2.1 looks at the monitoring problem in order to understand how to best *formalize* and *specify* compliance languages (which may take into account runtime information); D5.1 looks at the monitoring problem in order to understand how to best *capture* events, how to *store* them in a data warehouse, how to process them and how to *present* them to the human user.

1.2. Document overview

This document is structured as follows. In Section 2 we provide our general view on compliance and compliance management and we contextualize the role of monitoring in the overall compliance management life cycle. In Section 3 we briefly discuss the general offer of monitoring software, while in Section 4 we narrow our focus to service and service compositions (business processes), and overview state-of-the-art products and research approaches. In Section 5 we overview state-of-the-art business intelligence and reporting suites, which can be used for the visualization of the monitoring outputs in WP5. In Section 6 we analyze the results from the previous sections and specifically consider the case of monitoring for compliance management in COMPAS, providing an outlook of possible developments. Finally, in Section 7 we conclude the deliverable.

1.3. Abbreviations and acronyms

API	Application Programming Interface
ADP	Automatic Data Processing
BAM	Business Activity Monitoring
BI	Business Intelligence
BPM	Business Process Management
BPQL	Business Process Query Language
CMDB	Configuration Management Database
CMMI	Capability Maturity Model Integration
COMPAS	Compliance-driven Models, Languages, and Architectures for Services
CRM	Customer Relationship Management
EAI	Enterprise Application Integration

ERP	Enterprise Resource Planning
ETL	Extract, Transform and Load
FOL	First Order Logic
GUI	Graphical User Interface
HP	Hewlett-Packard
IT	Information Technology
KPI	Key Performance Indicator
OLA	Operational Level Agreement
QoS	Quality of Service
RBSLA	Rule Based Service Level Agreement
SLA	Service Level Agreement
SLO	Service Level Objective
SLM	Service Level Management
SOA	Service-oriented Architecture
UI	User Interface
XSAL	XML Service Assertion Language
WfM	Workflow Management
WS-BPEL	Web Services Business Process Execution Language
WSCoL	Web Service Constraint Language
WSLA	Web Service Level Agreement

2. Compliance management and the role of monitoring

In order to better understand the role of the monitoring activity in particular in WP5, we contextualize the activity in what is our interpretation of the overall life cycle for compliance management. Figure 1 depicts the compliance management life cycle, which touches all of the research issues addressed in COMPAS. Next, we briefly describe the role of the four phases we identify, i.e., Internalization, Design, Business Execution, and Evaluation, and the role of the Monitoring, Enforcement, Re-engineering, and Policy Adjustment activities.

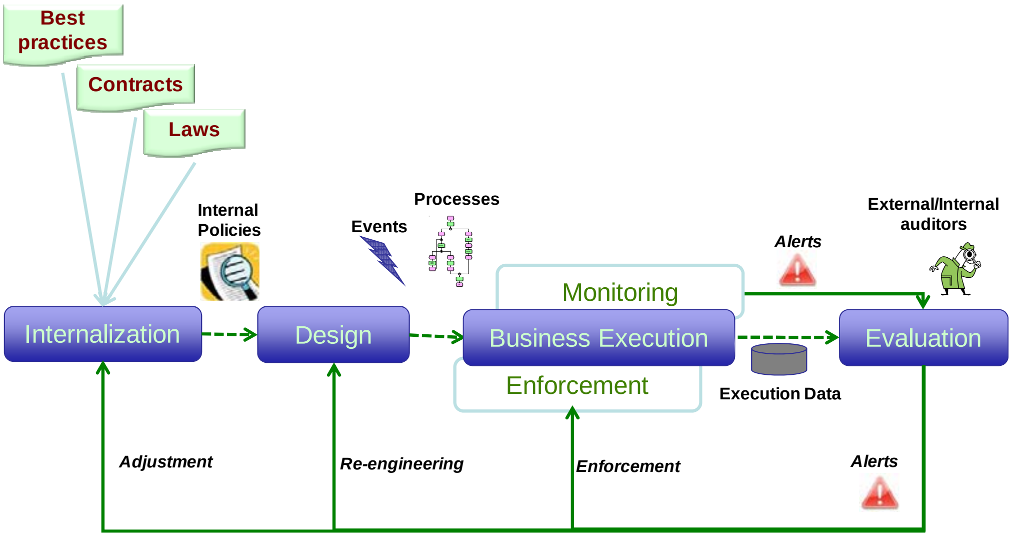

Figure 1 The role of monitoring in the compliance management life cycle. Solid arrows represent communications of events; dashed arrows represent data/product flows.

2.1 Internalization

In order to understand how to assure compliance for its business, a company needs to understand which compliance concerns really affect its business. It is important that a company understands the compliance regulations it must (or wants) to comply with, and that it identifies those regulations that really need to be implemented in the company. We call this activity *internalization* of compliance concerns. Internalizing compliance concerns means interpreting them and translating them into so-called internal policies, which collect all the compliance concerns the company must obey. Typical compliance regulations are, for instance:

- *Laws* are given by the government, typically specific to individual sectors. For instance, the Sarbanes-Oxley Act (SOX) of 2002 mainly focuses on auditing, the Health Insurance Portability and Accountability Act (HIPAA) focuses on protecting health information.
- *Best Practices* typically come from professional organizations and provide industry-specific methodologies and solutions. For example, Capability Maturity Model Integration (CMMI) is a best practice that describes characteristics of effective software development processes.
- *Contracts* are legal bindings among business partners, which have been agreed upon jointly. For instance, an Internet service provider might sign a contract regarding service level agreements (SLAs) with its customers.

Note that the above are just examples of the most representative resources of compliance regulations; the list is not exhaustive, and there might be a number of additional sources of compliance concerns, which might be necessary to take into account.

2.2 Design

Once the exact regulations that need to be implemented in the company are known, the ne

xt step is to design a compliant business. Compliantly designing business practices means specifying business processes by taking into account the identified (internal) compliance policies and relevant events (e.g., high-level business events or low-level execution events), which can be used to understand the state of a running process and to communicate predictable (known at runtime) compliance problems. In practice, designing for compliance means (re)structuring business practices in such a way to assure compliance by design, that is, assuring that the correct execution of a designed business is compliant and, possibly, that incorrect executions can be repaired to remain compliant. There are different forms the outcome of this design phase may have, ranging from verbal agreements or commitments and simple checklists on paper, to formally specified business processes in an executable language that can, for instance, be interpreted by a process or workflow engine. Depending on the company's business practices, the right means needs to be chosen.

2.3 Business execution

After the design phase, business is executed according to the specified business processes. It is important to note that the execution of the actual business is not necessarily subject to any automated support. That is, specified business processes may be supported by automated instruments such as customer relationship management (CRM) systems, enterprise resource planning (ERP) systems, workflow management (WFM) systems, or similar, just as they also could be executed in a fully manual fashion. For instance, it might be possible to fully automate an emailing service, but the delivery of regular mail might still be performed in a traditional, manual fashion; both delivery scenarios could, however, obey the same conceptual business process. The typical business setting, however, is represented by a semi-automated execution of business, where parts of the business are supported by some of the above systems and other parts are performed manually.

As discussed next, WP5 will specifically focus on the monitoring of the execution of compliant business processes and constituent services, according to the specifications deriving from the design phase.

2.4 Monitoring

While the specification of compliant business processes in theory assures compliance in the business execution, in practice the real business execution is always characterized by unpredicted problems that may lead to non-compliant outcomes or behaviors. The human factor but also hardware or software failures may cause deviations from the expected flow of work, which need to be captured, analyzed, and compared with the actually expected business process, so as to assess whether the deviation does or does not impact business compliance and whether countermeasures are required. It is the monitoring activity that enables this control.

As shown in Figure 2, the monitoring activity typically captures events generated during business execution (according to the events specified in the design phase). If parts of the business execution are automated, e.g. with the help from CRM, ERP, WFM systems, events will be generated by the process engine, in charge of running the process specification. The parts which are not automated may however generate user-driven events, such as notifications or emails, which can be captured to derive the status of the business execution. Here we do not make any further assumption about how events are generated, formulated, transmitted, or captured.

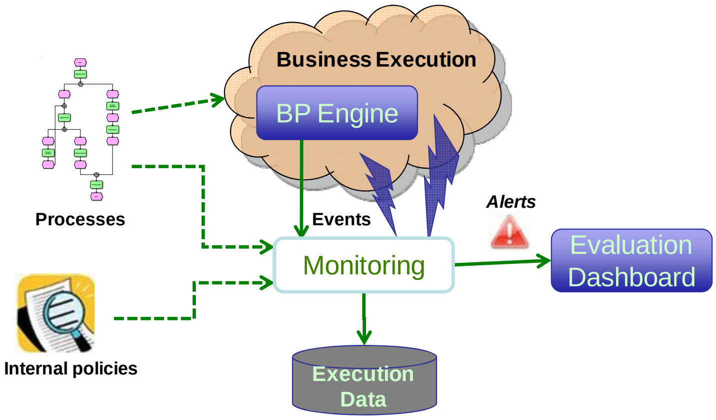

Figure 2 Monitoring business execution events.

Captured events, as shown in Figure 2, can be used to feed a monitoring dashboard with up-to-date information for human consumption or to log process execution data for data warehousing and later analysis.

2.5 Evaluation

The evaluation phase, finally, checks whether the monitored business execution complies with the specified, compliant business processes and the internal policies. Such evaluation might be partly automated (the computation of compliance-related indicators), but it is also under human control (e.g. the interpretation of computed results and reports). WP5 will particularly focus on the development of business intelligence solutions for compliance control, providing a compliance governance dashboard to so-called internal auditors (company-intern people, responsible to control and ensure compliance). The graphical visualization of events, alerts, and non-compliance problems will allow the internal auditor to assess the severity of the raised problems and, if necessary, to enforce countermeasures, re-engineer design artefacts, or adjust internal policies, in order to improve business compliance. More precisely, the dashboard will allow the issuing of queries over monitored runtime data for generating compliance control reports and discovering models for real process executions starting from the monitored events. Reports and analyses are generated or, respectively, performed periodically (e.g. each night); runtime information (e.g. events) might be published directly. The output of the evaluation phase is particularly intended for human consumption and not for automatic compliance enforcement; results published in the dashboard need to be interpreted and assessed by the human user, who, in response to the raised problems, will understand how to mitigate them.

External auditors, e.g. financial auditors, may also benefit from the availability of a central point of collection and control of all compliance-related concerns in the company. The dashboard immediately communicates which concerns are being controlled by the company, how many problems have been registered, and how they have been solved or approached. Of course, external auditors will also control conventional documents and will not limit their investigation to the dashboard only.

2.6 Enforcement, re-engineering, policy adjustment

In case non-compliance problems are detected, the typical countermeasures that can be enacted in order to assure compliance or mitigate possible negative effects, the following reactions may be performed:

- *Enforcement of automatically controllable actions*: In some cases, in response to raised events it might be necessary to timely enforce a compensation or correction action. For instance, if a deadline for the submission of a deliverable has not been respected, a possible enforcement action could be to send an alert email to the responsible author of the deliverable, urging him/her to produce the expected document as soon as possible.
- *Re-engineering of the design for compliance*: A simple enforcement action as reaction to registered events might not be enough to solve the non-compliance problem. This is for example the case, when a business process has been designed in a non-compliant fashion in the first place; even a correct execution of such a process will inevitably lead to a non-compliant result. In such cases, the initial design of the business process needs to be re-engineered.
- *Adjustment of the internal policies*: Finally, non-compliance problems might also be due to the wrong interpretation and, hence, the wrong internalization of compliance regulations, or simply because regulations change over time. In such cases, the policies need to be adjusted, and the business execution might require a re-engineering.

Note that enforcement, re-engineering, and policy adjustment are outside the scope of the WP5. We mention them here for the sake of completeness of the discussion, and to show how the output of WP5 will create compliance awareness and allow a company to react to possible violations.

3. Generic software support for monitoring

In general, there is a myriad of monitoring solutions that are able to provide insight into properties, i.e. metrics, of a running software system. Depending on the domain of the monitoring application, such parameters, the techniques adopted for their measuring and simply the purpose of the monitoring application may vary. Typical domains of monitoring applications are for instance:

- *System*: system monitoring tools focus for example on parameters like CPU load, memory usage, disk usage, system logs, network usage.
- *Network*: network monitoring tools focus for example on network services (protocols) such as HTTP, HTTPS, SNMP, FTP, SMTP, POP3, IMAP, DNS, SSH, TELNET, SSL, TCP, ping, but also bandwidth or throughput metrics.
- *Application*: application monitoring tools focus for example on faults, performance, configurations, security and accounting data of applications running inside an application server.
- *QoS* (quality of service): QoS monitoring tools focus are very common especially in tele-communications systems, where they focus for example on voice over IP connection quality.
- *SLAs*: SLA monitoring tools focus on typical network, system and application monitoring metrics, but also provide the possibility to automatically check compliance with SLAs and to generate events in case of violations.
- *Internet traffic*: Internet monitoring tools focus for example on email traffic, visited web sites, instant messaging services, web searches and similar; they are for instance used for child protection.

- *Viruses*: Anti-virus software monitors for example documents, applications, boot strap sectors, main memory, hard disk memory or generic drives in order to protect a PC against virus or similar.
- *Security*: Intrusion detection systems monitor for instance user interactions and network traffic to protect a system against unwanted access or manipulation of data (e.g. by crackers, malware such as trojans or worms, or employees).

Generally, the above tools are able to automatically generate events or send messages (e.g. via email, pager or SMS) in response to detected violations or if individual metrics go above or below predefined thresholds, in order to inform an administrator or the user of the detected problem or violation. Measured metrics, events and generated outputs are usually accessible via special presentation components (e.g. charts, dashboards, and reports) in the tools' user interface, which in most cases can be customized based on individual user preferences, so as to achieve desired management and monitoring perspectives (e.g. role-based or user-based). In some cases, plug-ins may allow the user to even develop own monitoring and management verification routines, according to individual needs and by using a programming language of choice (e.g. Java, C++, Perl, Ruby, Python, PHP or C#).

Keeping in mind the focus of the COMPAS project on web services and the SOA, in this deliverable we are particularly interested in web services, web service compositions and business processes. Regarding the compliance management life cycle model described in the previous section (see Figure 1), WP5 will thus focus on the monitoring of services and service composition executions (also encompassing some of the above features), in order to detect violations of compliance regulations (compliance governance). The following section thus devoted some more details to this problem.

4. Service and service composition monitoring

In this section, we focus on state-of-the-art service and service composition monitoring products and research approaches. While the products allow us to understand what features and what capabilities are relevant in the service monitoring context, the research approaches allow us to gain a deeper insight into how solutions are realized.

4.1. Service and service composition monitoring products

Looking at the current spectrum of industrial monitoring products that aim at assisting the management of service composition or business process executions is an important means to identify which are the current trends in the area and which is the state of the art regarding the supported features. The wealth of products and solutions available on the market indeed comes with a large amount and variety of different features, which is important to look at from different meaningful perspectives, in order to be able to understand common solutions and commonly accepted practices (ideally, best practices). In the following, we characterize such perspectives by means of so-called dimensions, which we then use to analyze prominent monitoring and service composition management tools.

4.1.1. Dimensions of analysis

Given that there are no readily available "compliance monitoring" solutions on the market, it is important to understand which of the currently available solutions provide practices or approaches that can be leveraged for the development of a monitoring solution for compliance governance. We therefore next introduce a set of analysis dimensions that allow us to look at the tools available on the market bearing in mind some specific requirements for a compliance governance solution. Specifically, we focus on six dimensions, i.e. Input, Output, Processing, Intrusiveness, Portability and Customizability. The goal of the selected dimensions is to understand what current products are looking at, what they are able to produce in output, what kind of business logic they are based on, how they influence a monitored system, whether they are of generic applicability and, finally, what kind of customization features they provide. We think these dimensions adequately capture both the common and the distinguishing features of the monitoring products, and that they are able to highlight what still needs to be done in order to effectively enable compliance governance. The dimensions are introduced in more detail in the following.

Input

The Input dimension encompasses *what* is monitored by the products in order to assist service composition management. In the specific case of compliance management, it is important to understand which is the exact object of the monitoring activity, in order to assure that a product effectively monitors all the relevant information that is necessary to support real business compliance management needs. We characterize the Input dimension by means of four properties, i.e., Type, Format, Access Mechanism, and Heterogeneity:

- *Type*: specifies which kind of information we would like to monitor by looking at the business or execution environment. For instance, typical types of information to be monitored are operations in a database, service invocations, message exchanges, network traffic, or events (real-time, business).
- *Format*: characterizes the format in which the input can be monitored. The format typically depends on the source of the input (e.g. database vs. web services). Common formats are for instance binary data, plain text, XML, or formatted data.
- *Access Mechanism*: describes how the monitoring instrument can access the input. We specifically distinguish between pull or push methods. The first require the monitoring solution to access and extract data autonomously. The second allow the monitor to listen for notifications automatically generated by the business or execution environment during the runtime.
- *Heterogeneity*: expresses whether the monitoring tool supports the monitoring of different input data sources in the business environment.

Output

The Output dimension describes *how* the results of the monitoring activity are provided in output to the users of the product interface, so as to aid decision-making. The Output characteristics are defined taking into account three properties, i.e., Presentation Components, Analysis Perspectives, and Specific Knowledge:

- *Presentation Components*: graphically render the results of the monitoring activity, in order to enhance and facilitate the visualization and comprehension of the output. For instance, typical presentation components are dashboards, reports, tables, charts, KPIs, and similar.
- *Analysis Perspectives*: represent the different views that one can have over the output of the tool, depending on to the users' roles, profiles, and objectives. The visualization of the output typically varies in its granularity (levels of summarization) of the data. For instance, different organizational roles such as managers, IT specialists, business practitioners may have a role-based view (e.g. considering access rights) of data at en-

gine level (to check e.g. availability, security), process level, service level, message level, etc..

- *Specific Knowledge*: expresses whether the output demands from the users any specific knowledge a part of the business environment, in order to understand and correctly interpret the output. Specific knowledge might, for instance, be programming languages (e.g. SQL), standards specifications (e.g. WSLA), or business rules.

Processing

The Processing dimension focuses on whether and how input data can be *transformed* into useful or meaningful information for service composition monitoring and management purposes and to support decision-making. This dimension, hence, considers the so-called intelligence part of the application. The Processing is articulated into three properties, i.e., Compliance Language, Data Analysis, and Simulation:

- *Compliance Language*: dedicated specification languages may allow the writing of contracts or monitoring rules, in order to configure the monitoring product to “understand” high-level regulations or compliance concerns. Typical compliance languages are for instance WSLA and RBSLA.
- *Data Analysis*: indicates how the product processes data to extract knowledge from logs, event traces, or metrics, e.g., providing root-cause analysis or trend analysis. For instance, the unified view over a service execution and the data correlation, in a central repository, make it possible to identify the root-cause of SLA violations, to provide a historical services performance perspective, to show summarized and detailed reports.
- *Simulation*: allows users to simulate an actual situation, to change service execution conditions and system performance parameters (e.g. to include hours variation in SLA schedules, and to simulate executions of service aggregation, when mistakes are made or required SLO data are missing, during initial SLA), and to verify the impact caused by these modifications in the business environment. This approach allows users to estimate the behavior of services and processes under various conditions, showing which parts of the system are impacted and enabling decision makers to take preventive actions (e.g. send warnings to avoid delivery delays, critical executions faults, or financial damages).

Intrusiveness

The Intrusiveness dimension expresses whether the monitoring activity introduces any *overheads* into the monitored system of business environment and, if it does, what kinds of overhead are introduced. Overheads typically impact on the execution performance of the monitored system. We characterized the Intrusiveness dimension by means of three properties, i.e., Response Time, Storage, and Network Traffic:

- *Response Time:* expresses if the monitoring product impacts on the response time of the monitored system (e.g. retrieval time of web services invocation). The response time delay introduced in a running system could be precisely measured in terms of seconds, minutes, or hours. However, in this deliverable we are just interested to know whether there is an impact on the system (or not) that could be perceived as inadequate by the users.
- *Storage*: states if the monitoring product requires a specific storage space. The Storage property does not exactly measure how much space is required by each of the tools; an

indicative value or an affirmative or negative answer suffices for the purpose of this survey (the reason for this is that without the availability of a real running system it is hard to precisely measure or assess the correct amount of space that is required). Indications of the storage space are thus based on information that is provided in product descriptions.

- *Network Traffic*: informs if the monitoring product causes some impact on the network traffic (e.g. consuming bandwidth).

Portability

The Portability dimension expresses possible software or hardware *requirements* that must be fulfilled in terms of integration and communicability, in order to run the monitoring product. The Portability of a product is mainly determined by three properties, i.e., Application Domain, Software Suite, and Operating System:

- *Application Domain*: states whether the monitoring product provides monitoring facilities for specific application or business domains (e.g. events related to financing or banking applications) or whether its applicability is general.
- *Software Suite*: tells if the monitoring product demands for the availability of a vendor-specific (proprietary) software suite to be installed and executed. That is, a monitoring tool might be able to operate only with an existing software suite, such as, for instance, IBM WebSphere or similar.
- *Operating System*: specifies if the monitoring product requires a specific operating system. Possible instances of operating systems are Windows, Linux, Unix, and MacOS.

Customizability

The Customizability dimension expresses whether the monitoring product can be *tailored* to specific user needs. In the particular context of monitoring products, it might be important to support user-defined input sources (to support input heterogeneity), output generation techniques, or data processing logics. Hence, this dimension concentrates on three properties, i.e., Input, Output, and Processing:

- *Input:* indicates whether inputs can be added or customized by the users to enhance the service monitoring support, e.g. according to new user or business needs. Both are essential to cope with business environment evolutions. Typically, products allow the addition of new SLAs and input parameters (e.g. metrics collecting a time interval, compliance periods).
- *Output:* states whether it is possible to define new output results or presentation components, to better assist specific needs. Tailoring the output may be crucial in order to support newly added inputs (e.g. KPIs to evaluate new SLAs) or users' necessities (e.g. addition of different charts to show metric values based on particular rules or target goals adopted by the company).
- *Processing:* specifies whether users can tailor the processing of the service monitoring product. For instance, users may insert a new mathematical or logical expression to achieve output results that better fit their necessities, or create simulations to analyze expected service performance.

4.1.2. Product descriptions

Next, we discuss a set of products in more detail, which represent – to the best of the authors' knowledge – a representative snapshot of the current practice in service and service composition monitoring. We have selected products from main players on the marked (e.g. IBM, HP, Oracle, and Mercury), also based on their successful adoption in renowned companies (e.g. Vodafone, Nokia, Siemens Business Services, and so on). Each of the tools is briefly described and evaluated by means of the analysis dimensions defined above:

- BMC Service Level Management (http://www.bmc.com/BMC/Common/CDA/hou_Page_Generic/0,3465,81909862_92306903,00.html);
- HP Open View Service Navigator (http://h20229.www2.hp.com/products/servnav/ds/servnav_ds.pdf);
- IBM Tivoli Service Level Advisor (http://www-306.ibm.com/software/tivoli/products/service-level-advisor/);
- HP/Mercury Business Availability Center (https://h10078.www1.hp.com/cda/hpms/display/main/hpms_content.jsp?zn=bto&cp=1-11-15-25_4000_100__);
- Unicenter Service Assure (http://ca.com/products/product.aspx?id=4573);
- NimBUS (http://www.nimsoft.com/solutions/index.php);
- Oracle Business Activity Monitoring (http://www.oracle.com/appserver/business-activity-monitoring.html).

BMC Service Level Management (SLM)

The BMC Service Level Management (SLM) 7.0 product combines the service support, infrastructure metrics and events data into a common SLM product, which allows users to set agreements and service target goals. The goal of the SLM product is to allow analysts to track business performance and availability targets of their infrastructure components and service desks, which gives to the business a high-level, detailed and real-time picture of where problems exist. So, they can correct them and try to maintain high-quality service. The three entities used to monitor the service levels are: service targets, agreements, and contracts. All of them are created based on the wizard interface, without any hand coding. The SLM solution has been applied in several success cases, for instance, Vodafone, Diageo, and Capital District Physicians Health Plan.

The SLM raw input data is collected from specific collection points by a Java API executed as a web service and published with Apache Axis. The SLM input data type can be represented by numeric or status variables, which express availability and performance of the machines, services, and applications from the business environment. The SLM just allows data extraction from products developed by BMC (e.g. BMC Remedy AR System, Patrol, ART, and SIM). The SLM output results can be visualized by dashboards (two types, i.e., SLM, customer), Crystal reports, KPI charts, and view forms. The final information also can be expressed by means of different analysis perspectives (e.g. company, organization, department, and supplier). No specific knowledge, apart from business domain expertise, is required to interpret the output results. The product's documentation does not mention any special compliance language to process the service monitoring data input, but it refers to a special method that provides impact cost analysis for a future period, when some service target is not met. On the other hand, no comment or explanation was found in it about simulations features, response time and network traffic intrusiveness. However, this documentation mentions about the AR System Repository that is responsible to store all the input data extracted, as well the output results. Considering portability properties, the SLM can be executed in Windows or Unix platforms and is totally dependent from the others BMC products, using them as input data sources. Finally, the SLM product offers some specific customizability features to add agreements, contracts and templates, and to configure, insert or exclude targets, KPIs or alarms. In addition, some special processing features also are available, like defining new: collection points, collection nodes, arithmetic and Boolean expressions.

HP OpenView Service Navigator

The HP OpenView Service Navigator is a real-time and top-down solution to manage applications and services from the business perspective, and to learn of the business impact of lower level component failures or performance degradations on the whole business environment. This product presents an easy-to-use graphical user interface to manage the complete life cycle of the operational service views (e.g. create, test, and deploy), providing less complexity and greater cost efficiency. However, HP announced the OpenView Service Navigator discontinuance in December 31st, 2005. However, due the relevance of this product and its pioneering role in linking the business with the IT infrastructure it relies on, we believe that it should be mentioned in our survey. Besides that, some reputed companies have been adopted the HP solution to manage their SLM, e.g. Nextenso, Nokia, TANTAU [HP00], and Con Edison Communications Assure [HP03]

The OpenView input data is based on end-to-end services logs environments (e.g. network elements, computer systems, databases and applications). Typically, the log values comprise metrics from CPU utilization, process queue lengths, used network bandwidth, memory utilization, and swap rate. These metrics usually have numeric and character format and are collected inside the Utility Data Center by OpenView. For that reason, this product does not support heterogeneous input. The results from the management and monitoring product can be visualized by the users through graphical views and reports, by means of different analysis perspectives (e.g. business organization levels or roles, geography localizations, business application logic, or any other categorization). The users do not need any special computation knowledge to obtain or understand the output results. Among the output data, the OpenView provides impact and root-cause analysis. This solution also allows the user to execute simulations of services executions, which, for instance, helps estimate the impact of events within the IT infrastructure, using the defined propagation and calculation rules. The outputs of this commercial product are achieved based on some processing activities executions, but the HP OpenView documentation does not clarify how they are done, their response time amount and their network traffic consume. On the other hand, the HP documentation explains that all the data (i.e. input, output and processing) are stored in a central repository running with Oracle DB or MS SQL2000. The OpenView has a high portability level, presenting completely operating system independence and no software suite dependence, allowing seamless integration with other services execution and management tools. In terms of customizability, this product presents a graphical user interface (GUI), where the users can configure important aspects of a service, without any knowledge about the internal configuration solution details or about the programming language adopted. Such aspects encompass mainly rules addition as input data, services views and reports like output results.

IBM Tivoli Service Level Advisor

The IBM Tivoli Service Level Advisor provides real-time SLM capabilities for companies that need to measure, manage, and report on availability and performance aspects of their internal IT infrastructure to the users. Tivoli Service Level Advisor collects performance and availability metrics and compares them with service level objectives (SLOs) and its SLM capabilities complement the performance and availability measurement functions of other Tivoli products, e.g. IBM Tivoli Monitoring for Transaction Performance and IBM Tivoli Business Systems Manager. The business goals of the Service Level Advisor are: provision of SLAs meaningful to businesses, automation of SLA report production to reduce costs and provide timely report delivery, provision of a mechanism to resolve disagreements on SLA achievement, and provision of early warning of trends toward SLAs being breached. The ADP (Automatic Data Processing) to keep corporate payrolls running [Ptak07] can be mentioned as one of the Tivoli Service Level Advisor solution success stories.

This IBM product works with metrics (e.g. numeric and characters) from the existing monitoring and event correlation applications to provide output results data to the users. The Tivoli Service Level Advisor captures these data from a homogeneous software environment, where only IBM applications are available (i.e. Tivoli NetView Family, Tivoli Enterprise Console, and Tivoli Monitoring for Transaction Performance), using a specific extract, transform and load (ETL) process. Based on those input metrics, output results are produced and provided by means of dashboards, details (e.g. overall details, SLA results, trends and violations), ranking (e.g. SLA, SLA Type, Customer, Realm and Offering Component), summary reports in the form of tables and graphs, and web-based customer reports. All these presentation components are allowed to offer such information through different analysis perspectives, i.e., customer, executive, operator, and object type (e.g. SLA, SLA Type, realm, offering component and resource). No specific computational knowledge is mandatory to understand the output results. Among the latter, the IBM product provides exponential, stress detection algorithm, impact analysis, linear algorithms, root-cause analysis, and trend analysis. In addition, this product also supports costumer transactions simulation. The processing mechanisms used in order to produce the final results are not detailed by the IBM documentation. It just mentions that the Tivoli Data Warehouse, the SLM measurement data mart, and the SLM database are used to store information to monitor and to manage service compliance. About response time and network traffic consume no details can be found. Also according to the IBM documentation, the Tivoli Service Level Advisor presents a full integration and portability with IBM Tivoli NetView Family and the infrastructure for the service may consist of a set of applications running on Unix and Microsoft Windows 2000 servers. Tivoli Service Level Advisor offers the opportunities to create new inputs for offerings SLAs, and also to specify schedules and define peak times and other schedule states (e.g. standard, prime, off hours, and others) for varying levels of service, defining how often evaluation and trend analysis should be performed, determining breach values for metrics associated with offerings, and manage active SLAs.

HP/Mercury Business Availability Center

HP Business Availability Center is a real-time, comprehensive business service and application management solution, which allows monitoring the performance of business services and applications from the point of view of the consumers of those services – the business, its customers and its partners. The Center is composed by such HP set of products (e.g. Business Availability Center Software for Composite Application Management, Business Process Insight, Business Process Monitor, and Business Service Level Management). The set of products allows measuring business impact and risk from the end-user perspective, to manage business and operational service levels proactively, to accelerate problem isolation by automating standard operational processes, to manage complex business transactions across heterogeneous environments, and to manage the complexity of composite applications and SOA. For instance, the City of Boston Web-based applications can be mentioned as one of the Business Availability Center success stories.

The HP Center inputs deals with transactional patterns and transaction content patterns, monitoring them to identify issues and trigger automated processes or notifications across all tiers in the enterprise, including J2EE application servers, messaging middleware (e.g. WebSphere MQ and Java Message Service - JMS implementations), and mainframe transaction monitors (e.g. Customer Information Control System - CICS and IP Multimedia Subsystem - IMS). Moreover, HP Center allows to define, track and manage service levels to meet business objectives and measure system service levels for availability and performance, and map to business services for managing equivalent infrastructure-centric SLAs. Typically, those metrics present numeric and characters format, and are collected from a homogenous environment, where just HP applications are adopted to provide services monitoring and management information. The HP Center output results can be visualized though dashboards and reports, which contain impact analysis, change tracking and performance by transaction. Such reports present high-level summaries as well as detailed data on specific activity for select periods of time. All those results are organized according to three analysis perspectives: role-based, user-based, and customizable, and they do not demand special computational knowledge. Among the results the HP Center provides trend and root-cause analyses, but any simulation feature is offered. The HP Business Availability Center official documentation does not present any specific comment about the compliance language, the time response and the network traffic consumed by the solution. The document just mentions that the data used during the solution processing is stored into the HP Universal CMDB and is dependent from HP's software (e.g. Composite Application Management and HP Business Process Insight) and Windows operating systems. The Business Availability Center has some customizability features that enable users to create baselines in order to establish realistic SLOs for availability and response times for the different subsidiaries, geographies or organizations they serve, and also realistic, quantifiable availability and performance objectives that reflect business goals. The product also provides features to create personalized views from dozens of pre-defined components, enabling users to focus on the KPIs for critical business services.

CA Unicenter Service Assure

The CA Unicenter Service Assure translates IT metrics into manageable SLOs and KPI, and provides pre-defined and custom reports for communication of infrastructure services to IT operations and management. Beyond that, this product aggregates and analyzes data from various resources and applications, monitoring contracts and SLAs in real-time and reporting of user service levels against SLA parameters, warnings of pending violations and prioritization of performance issues based on end-user and business impact. It also allows running what-if simulations to test temporary failures in the critical business services. Krishak Bharti Cooperative Limited is one example of a success story for the Unicenter Service Assure adoption.

The common inputs used by the Unicenter Service Assure are disparate resources, applications data, and business requirements documents for IT services, which are automatically translates into a set of SLAs. Typically, these inputs present numeric, characters, and plain text format. Unicenter Service Assure collects inputs from a homogeneous environment composed by CA's Enterprise IT Management (EITM) framework. Regarding the outputs the product provides dashboards, colours-code charts, and reports, which contain financial administrative information. Such results can be visualized by means of two main role-based analyses perspectives, i.e., IT and business, without any computational specific knowledge. The Unicenter Service Assure provides performance reporting capabilities, root-cause and trend analyses, and one special feature to execute simulations (e.g. to test the impact of an outage on your critical business services). All the data used by this solution are stored in a central data repository with security access control. The Unicenter Service Assure documentation does not explicit any information about the compliance language adopted, the response time and the networks traffic consumed by its solution. However, the CA solution runs installed in Windows or Linux platforms and permits some features customizations like SLAs addition (input), report service level goals (output), and performance goals and simulations. The latter allows testing the impact of an outage on your critical business services (processing). For instance, if Unicenter Service Metric Analysis identifies a bottleneck that may result in IT service slowdowns or disruptions, you can open the monitoring program to quickly discover the contractual service level and the penalty for an SLA violation [CA07].

NimBUS

The NimBUS provides a real-time solution for monitoring the performance and availability of the IT infrastructure, both physical and virtualized, based on KPIs of business service processes. In order to allow SLA monitoring, NimBUS contains templates to insert the SLA definitions, data analysis infrastructure, SLA compliance calculation and breach forecasting with warning alerts, automate SLA report generation and distributions, with historical trend analysis. Beyond that, NimBUS offers investments protection with 3rd party tools data integration, and service desk integrations to monitor Operational Level Agreements (OLAs). The NimBUS solution is adopted, for instance, by Siemens Business Services AS, University of California, and MTU Aero Engines GmbH.

Nimbus suite product gathers special features to monitor data from: applications (e.g. SAP, VoIP, .NET, and J2EE), user response time, servers (e.g. Sun Solaris, IBM AIX, IBM iSeries, HP HP/UX, and Red Hat), databases (e.g. Oracle, Sybase, DB2, and MS SQL Server), and network (e.g. network traffic, SNMP, DNS, and DHCP). Typically all these input data are collected (pull) and represented by numeric, characters and plain text format, considering a heterogeneous environment of services monitoring and management tools. The Nimbus output results can be visualized through dashboards and reports, which provide representations of both IT and business service KPI's. The reports have automated web-based generation and provide historical SLA compliance coverage. Those results are presented according to two main perspectives: IT and business, and can be accessed along a time line, based on drill-down operations on past periods and status, without any computational specific knowledge. Among the results the NimBUS solution provides historical analysis (i.e., reporting is the ability to modify and recalculate past SLA periods), mathematical formulas (e.g. interval, median, and average) to analyze and summarize QoS data points, root-cause and trend analyses. On the other hand, no simulation feature is offered by this solution. Nimbus documentation does not explicit the compliance language adopted, the response time and network traffic consumed by its solution. NimBUS has a central database, where the role data used to produce the output results is consolidated and runs on iSeries AS400, Netware, Linux, Windows, and Unix. The implementation of NimBUS encompasses special feature to treat SLA input parameters (e.g. compliance period, operating period, exclude periods, compliance percentage, and compliance calculation methods) and to build tailored outputs by means of dashboards views and graphical SLAs templates.

Oracle Business Activity Monitoring (Oracle BAM)

The Oracle Business Activity Monitoring (BAM) supports the ability to instrument ERP systems, business applications, legacy systems, and business processes from a company and to monitor their business events. BAM is also able to correlate business events with each other based upon user-defined rules. It also provides means to understand the impact of events on KPIs, which measure the business performance, in real-time. BAM provides the operational managers with a versatile process monitoring and analytics tool that can help them better analyze business process information in real-time by computing the higher-level complex event aggregates, thresholds, identifying causal relationships between different event types, complex temporal event patterns, and root-cause behavior identification. BAM expectation is to monitor the business state in a non-disruptive way and instrument existing structured, semi-related business processes without changing the way running processes are orchestrated and executed. Oracle BAM is adopted, for instance, by monsters.com [Barlas06], Metro Group [Oracle06a] and AR Telecom [Oracle06b].

The BAM solution communicates and extracts input data easily from existing production applications, business process management (BPM) tools, Enterprise Application Integration (EAI) system, JMS queues, and applications that communicate via web services. Thus, BAM's heterogeneous input set can pull data from event traces, data warehouses, messages, business processes, operational data stores, services, and XML sources. All these data can be represented as numbers, characters or plain text. Like traditional products, BAM also has special presentation components to provide its outputs in a dashboard; it supports proactive and instant alerts, streaming data delivery and reports, which contain, among others, charts, cross tab, spreadsheets, KPIs, and tables. All information is presented in an intuitive browser-based user-interface, accessible via multiple devices, driving enterprise-wide availability of real-time information. This product offers different analysis perspectives, according to its end user need, i.e., business executives, operation managers, responsibilities, roles, and skills of each user. It does not require any special computational knowledge to access and understand the results output by BAM. Regarding the processing of input data, BAM in particular focuses on event correlation and root-cause analyses. It is rather hard to assess the Intrusiveness of BAM; provided that the monitored applications and systems are able to generate the necessary events, the instrumentation of business processes can be done entirely inside the BAM environment, without impacting on the process executions. Besides the events and rules, no dedicated compliance language is supported. In summary, BAM can easily be integrated with production applications, BPM tools, EAI system, JMS queues and applications that communicate via web services. Despite this flexibility, this solution only runs on Microsoft Windows Server as operating system. Collected data are stored in a central repository named Oracle BAM Active Data Cache, which enables BAM to obtain complex temporal event patterns, to perform event processing and to run root-cause analyses. Finally, as for the customizability of the BAM solution, rules can be added/excluded or modified by the users, and presentation components like alerts, dashboards and reports, and security levels can be configured by the user. Besides that, the user also can customize data flow plans, data objects, rules and metadata.

4.1.3. Summary

Table 1 to Table 5 summarize the previous discussion. The tables and the above overview show that, despite the existence of a variety of mature monitoring solutions in the field of services and service compositions, we are still far from a comprehensible solution for compliance monitoring and management, i.e. from compliance governance. In Section 3 we have shown that there are monitoring solutions that focus on concerns such as system state, network traffic, applications, QoS, SLAs, Internet traffic, viruses and security; in this section we have particularly focused on services and service compositions. What is missing today, is the capability to process and interpret *generic events* in the monitoring tools, events such as user-defined business events or – more importantly in the context of COMPAS – compliance-related events. The capability to handle generic business or compliance events as first class citizens in the monitoring process, in the transformation of the captured events and in the computation of final reports would greatly extend the applicability of the examined solutions to the context of COMPAS. Oracle BAM is slightly going into this direction, allowing the user or administrator to define customized events and rules; the syntax of the available events and rules is however very limited and not yet flexible enough to cater for the particular needs of compliance management.

Most of the examined tools support the definition of thresholds for parameters or SLAs to be monitored and the possibility to generate events in response to a violation of a threshold or SLA. More *expressive languages* are however missing, for instance, languages that would be required to express complex event correlations and higher-level compliance rules. Again, Oracles BAM provides some features that could be used in this context (e.g. the event correlation mechanism); however, real compliance concerns cannot be adequately expressed and, hence, monitored. Also, in this regard, the ability to *compare* monitored business process executions or, more in general, business patterns with expected execution behaviors is not yet supported.

However, it must be noted that the above discussed tools represent mature solutions in the context of service monitoring, NimBUS being the most complete solution, Oracle BAM being the most promising solution as for what regards compliance management. In general, all the tools go far beyond the actual monitoring (capturing) task and also come with some form of (internal) *data warehouse* for the persistent storage of monitored information and with more or less flexible *customization facilities* that allow users to tailor the data processing and the report generation, without writing any own code. Especially what regards the presentation of results in the user interface, all tools are characterized by advanced, graphical *monitoring dashboards* that allow the interactive inspection of current (real-time) and historical data without requiring any specific training in addition to the necessary business domain knowledge. For the processing of monitored data, it is worth noting that almost all of the tools provide *root-cause* and *trend analysis* features, while some also support *simulations* features.

Table 1 Summary of the input dimension.

		BMC SLM	HP Service Openview Navigator	IBM Tivoli Service Level Advisor	HP/Mercury Business Availability Center	CA Unicenter Service Assure	NimBUS	Oracle BAM
Input	*Type*	• Data about availability and performance of machines, services, and applications (e.g. collection points) • Service targets, agreements, and contracts	• Applications and services • End-to-end service environment logs (e.g. network elements, computer systems, databases and applications) • Metrics (e.g. CPU utilization, process queue lengths, used network bandwidth, memory utilization and swap rate).	• Applications • Databases • Events • Web servers	• Applications servers • Availability and performance metrics • Mainframe transaction monitor • Messaging middleware • SLA metrics (e.g. KPIs - volume of users and mean time to repair - MTTR)	• Business requirements documents • Disparate resources and applications data	• Applications data (e.g. SAP, NET, and VoIP) • Databases data (e.g. Oracle, Sybase, DB2, and MS SQL Server) • Network availability and performance data (e.g. switches, network traffic, SNMP, and DNS) • Server data (e.g. Sun Solaris, IBM AIX, and Red Hat)	• Event data • Data warehouses • Messages • Monitoring business processes • Operational data store • Services • XML sources
	Format	Numeric and status	Numeric and Characters	Numeric and Characters	Numeric and Characters	Numeric, Characters, and Plain Text	Numeric, Characters, and Plain Text	Numeric, Characters, and Plain Text
	Access Mechanism	Pull	Pull	Pull (ETL)	Pull	Pull	Pull	Pull
	Heterogeneity	Yes (e.g. BMC Remedy AR System, Patrol, ART, SIM, etc)	No, just capture data from Utility Data Center	No (IBM Applications)	No (HP Applications).	No (CA's Enterprise IT Management -EITM framework)	Yes (many applications, servers, networks, and databases)	Yes (multidimensional and relational data sources, web services, enterprise application data)

Table 2 Summary of the output dimension.

		BMC SLM	HP Service Openview Navigator	IBM Tivoli Service Level Advisor	HP/Mercury Business Availability Center	CA Unicenter Service Assure	NimBUS	Oracle BAM
Output	*Presentation Components*	• Crystal reports • Dashboards • KPI • View forms	• Graphical views • Reports	• Dashboards • Details • Ranking • Summary reports in the form of tables and graphs • Web-based customer reports	• Dashboards • Reports	• colors-coded charts • Dashboards • Reports (financial administration)	• Dashboards • KPI • Reports (e.g. SLA compliance historical, with trend analysis, automated web-based generation) • Status indicators (e.g. current SLA period, current SLA compliance in period)	• Dashboards • Proactive and instant alerts • Reports (e.g. charts, columnar, cross tab, spreadsheets, KPIs, and lists) • Streaming data delivery
	Analysis Perspectives	• Company • Department • Organization • Supplier	• Business application logic • Business managers • IT managers • Executives • Geography • Operators • Organization	• Customer • Executive • Object type (e.g. SLA, SLA Type, customer, realm, offering component and resource) • Operator	• Customizable • Role-based • User-based	• Role-based	• Business • IT	• Business executives • Operation managers • Responsibilities • Roles • Skills of each user
	Specific Knowledge	No	No	No	No	No	No	No

Table 3 Summary of the processing dimension.

		BMC SLM	**HP Service Openview Navigator**	**IBM Tivoli Service Level Advisor**	**HP/Mercury Business Availability Center**	**CA Unicenter Service Assure**	**NimBUS**	**Oracle BAM**
Processing	*Compliance Language*	-	-	-	-	-	-	-
	Data Analysis	• Impact cost analysis for a feature period, when some service target is not met	• Impact analysis • Root-cause analysis	• Exponential; stress detection algorithm • Impact analysis • Linear algorithm • Root-cause analysis • Trend analysis	• Root-cause analysis • Trend analysis	• Performance reporting capabilities • Root-cause analysis • Trend analysis	• Historical analysis (i.e., reporting is the ability to modify and recalculate past SLA periods) • Mathematical formulas (e.g. interval, median, and average) to analyze and summarize QoS data points • Root-cause analysis • Trend analysis	• Complex temporal event patterns • Event processing technology • Root-cause analysis
	Simulation	-	Status simulation (e.g. helps estimate the impact of events within the IT infrastructure using the defined propagation and calculation rules)	Costumer transactions simulation	-	Simulations (e.g. to test the impact of an outage on your critical business services)	-	-

Table 4 Summary of the intrusiveness and portability dimensions.

		BMC SLM	HP Service Openview Navigator	IBM Tivoli Service Level Advisor	HP/Mercury Business Availability Center	CA Unicenter Service Assure	NimBUS	Oracle BAM
Intrusiveness	*Response Time*	-	-	-	-	-	-	-
	Storage	Yes (AR System)	Yes (installed in Oracle DB or MS SQL2000)	Yes (Tivoli Data Warehouse, SLM measurement data mart, and SLM database)	Yes (HP Universal CMDB)	Yes (Unicenter Service Assure database)	Yes (QoS)	Yes (Oracle BAM Active Data Cache)
	Network Traffic	-	-	-	-	-	-	-
Portability	*Application Domain*	-	-	-	-	-	-	-
	Software Suite	Yes (BMC Products)	No (Seamless integration with other tools)	Yes (IBM Tivoli NetView Family, e.g. IBM Tivoli Enterprise Console)	Yes (e.g. Composite Application Management and HP Business Process Insight software)	-	No	No (production applications, BPM tools, EAI system, JMS queues and applications that communicate via web services)
	Operating System	• Unix • Windows	• Independence	• Unix • Windows	• Windows	• Linux • Windows	• iSeries AS400 • Linux • Netware • Unix • Windows	• Windows Server Intel x86 versions

Table 5 Summary of the customizability dimension.

		BMC SLM	HP Service Openview Navigator	IBM Tivoli Service Level Advisor	HP/Mercury Business Availability Center	CA Unicenter Service Assure	NimBUS	Oracle BAM
Customizability	*Input*	• Agreements • Contracts • Penalties for noncompliance • Services • Templates	• Rules	• Offerings • SLA	• Baselines (realistic SLOs for availability and response times for the different subsidiaries, geographies or organizations) • Realistic, quantifiable availability and performance objectives	• SLA	• SLA parameters, (e.g. compliance period, operating period, and much more)	• Rules
	Output	• Targets • KPIs • Alarms • Templates	• Service views • Reports	-	• KPIs	• Report service level goals	• Graphical SLA templates • Dashboard views	• Alerts • Dashboards • Reports • Security levels
	Processing	• Collect points • Collect nodes • Arithmetic Expressions • Boolean Expressions	-	• Evaluation and trend analysis frequency; • Breach values for metrics associated with offerings.	-	• Performance goals	-	• Data flow plans • Data objects • Objects and rules • Metadata

4.2. Research approaches

In the previous section, we focused on industry solution, while here we turn our attention to academic research on monitoring. We describe some relevant approaches on monitoring business processes, security and privacy, and process mining from a research point of view. We will characterize the different approaches with regards to a set of dimensions of analysis.

4.2.1. Dimensions of analysis

The same remark as for the monitoring market solutions can be made for the research approaches: to the best of our knowledge there are no available compliance monitoring research approach. We will focus here on some research approaches selected according to the following three relevant axes for the COMPAS project: (1) business process, (2) security/privacy, and (3) process mining techniques. We focus on these axes because the research investigations within COMPAS will mainly concern these axes.

For surrounding research approach, we will consider the same dimensions of analysis presented in the section 4.1.1, excepting that we do not consider the intrusiveness dimension, because in the general case we do not have detailed information about the experimentation of an approach and its execution in real environment. Also, we do not consider customizability dimension, because most of the approaches do not reach a required degree of maturity to offer this type of feature and we will not investigate the interoperability dimension since most of the approaches we will describe deal with languages for specifying issues that will be monitored. Finally, we will introduce a new dimension "*Task*" which also encompasses the *Output* dimension. In general for each dimension we add new criteria, which in our opinion better fit to characterize research approaches.

Input

The Input dimension encompasses *what* is monitored by the research approach in order to assist service composition management. It is the same as input dimension presented above, but for the research approaches, we will focus more on the language that is used to represent the input rather than the used format (which can be seen as a physical property of the input). We believe these are very important properties for business compliance management systems. The reason is that compliance requirements will be expressed by different languages displaying different expressive powers. Also, compliance requirements vary depending on the type of data stored on the systems. This means that access mechanisms are language-dependent too.

- *Languages:* refers to the ways the monitoring properties are specified, the approaches may differ in the features of the specification language and are based on different formalisms and modeling notations. Logic-based notations allow for the specification of a property as a logical formula of a special form. Depending on the formalism and its expressiveness, the formula may characterize only a state of a system at a certain time (e.g., in approaches that monitor pre- or post-conditions of the service invocations) or express properties over an execution trace or its fragment (e.g. the approaches relying on temporal logics).
- *Type:* refers to the parameters used by the monitoring components. The parameters could be queries, events, etc.
- *Access mechanism:* refers to the way data are collected.

Task

The monitoring approaches can be classified according to the monitoring task supported by the approach; we characterize the task dimension by means of three properties, i.e., Goal, Aspect, and Output:

- *Goal:* the monitoring approach may address different aspects such as those relating to (i) *software fault detection:* provides evidence that a program behavior complies or does not comply with specified properties during program execution [DGR04]; (ii) *diagnosis*: gives information to users on identified faults that will aid in understanding the cause of the problem and may be help on; (iii) system *recovery*: consists in driving the necessary actions to return the system to the «normal» execution; and/or (iv) inspection of a program's behavior: measure, analyze, and report KPI values, their presentation in dashboards, automatic and proactive notification in case of deviations.
- *Aspect*: Indicates which aspect of monitoring the approach targets, e.g. security/privacy or behavior monitoring. Here we make a difference between security/privacy and behavior in the sense that the behavior is mainly related to the execution of a service, while security/privacy could be related to the way external agents interact with the service. For example, we might be interested in collecting a set of suspicious queries issues against a database.
- *Output:* Describes how the monitoring results are provided to the users (through an interface) or to the business execution environment. Most of research approaches are restricted to provide a simple output, e.g. alert message, graphics, text, and sending an event message.

Processing

For the Processing dimension, two different properties can be considered: Source of information, and Techniques:

- *Source of information:* indicates the sources of information used to extract data and relevant events. Typically, application logs, context sensors, message queues, process containers, and the corresponding data storages are the common sources of information. However, some approaches also defined ad-hoc information repositories as sources of information. For example, in **[**BJB+07] special reputation repositories are introduced and dynamically updated.
- *Techniques:* deals with the particular techniques adopted within the approach to perform the monitoring task. An interesting candidate is for example process mining, which is widely used for extracting patterns from process logs; the patterns describe the actual process and can be compared to the process specification. Another potential technique, for implementing monitors, is the aspect-oriented approach that allows embedding of the monitoring code, without affecting the running program and, in many cases, the platform code. Automated planning techniques are exploited in order to extract the monitor programs from the behavioral models of the involved services, and complex property specification **[**PT07], taking into account non-determinism and partial observability of the composition behavior.

Invasiveness

For the research approaches, we focus on the degree of invasiveness of an approach; this dimension expresses a way the monitoring framework is integrated with the application architecture and the way the monitors execute regarding the business environment. So, two properties can be considered, i.e., Architecture and Execution:

- *Architecture*: indicates a way the monitoring framework is integrated with the application architecture. In some approaches, the definition of the business logic and the monitoring activities are highly intertwined (e.g., through the use of annotations in the process definition). Other approaches keep the specification of the monitoring logic entirely separated from the business logic, thus encouraging a "separation of concerns", which allows designers to reason separately on the two problems [GG07].
- *Execution*: indicates a way the monitors are executed. This could be synchronously with the application code, blocking it until the property evaluation is done, or asynchronously in an independent parallel thread or even on another machine.

At this level, we can make the following difference between invasiveness and intrusiveness (used to characterize the industry solutions). Intrusiveness is mainly about execution parameters (time, storage and traffic) while invasiveness is about architecture configuration.

4.2.2. Approaches description

Business process monitoring

Monitoring business process address two kinds of problems, the first one is run-time requirements validation and monitoring, so it focuses on the problem of checking whether certain predefined properties are satisfied, when the system is executed. The next one, "Business Activity Monitoring" (BAM), deals with real-time monitoring of business activities, measurement of KPIs, their presentation in dashboards, and automatic and proactive notification in case of deviations.

Dynamic monitoring of WS-BPEL processes

The authors of **[**BG05] propose WSCoL Language for specifying constraints on execution by defining a set of monitoring rules. WSCoL provides the necessary constructs to define both functional and non-functional constraints and properties, with the capability of setting the degree of monitoring at run-time such as: validity time frame, priority, and set of certified providers for which monitoring may be omitted. Also, it enables specifying expressions over the process variables and supports set of built-in functions, logical and mathematical operators, and quantification.

In [BBG+07] the authors extend the work presented in **[**BG05] by considering the kind of properties the approach can monitor. The extended specification language, namely Timed WSCoL now allows for specifying temporal properties over the events that occur during the process execution.

The monitoring rules are deployed together with the process through weaving procedure, i.e., parse monitoring rules and add specific WS-BPEL activities to a process in order to achieve dynamic monitoring. At run-time the modified process interacts with the proxy service, a rule manager, which is responsible for processing the monitoring management instructions, processing monitor configuration, obtaining information from external data sources, evaluating monitoring expressions, and interacting with the actual services (instead of the original process). If some constraints are not met (monitoring rules are not satisfied), the monitoring manager will be responsible for informing the BPEL process about the enforcement.

Requirements monitoring based on Event Calculus

Authors in [MS05] consider monitoring web services as a problem of verification of requirements at run-time. This consists in checking if a process behaves in conformance with its specification, which states a set of behaviors that the process' services must exhibit during their enactment, as well as assumptions on these behaviors. In this work, all the behaviors and assumptions were stated as Event Calculus predicates, a logic-based formalism used for representing actions and their effects on some variables called fluent. Event Calculus encompasses some constructs that express complex situations, using time variables, universally or existentially quantified, and implications between predicates. The monitoring consists of checking the messages sent between the different services against the stated behaviors and assumptions. This needs a prerequisite transformation task of the business process specified in BPEL to Event Calculus predicates to be fully exploitable. An execution is said to be not conform, with regards to the specifications, when its entailed events are logically inconsistent with the behaviors or assumptions. The monitoring framework was implemented as a toolkit for monitoring service compositions specified in BPEL. The logs generated by the process engine were used to identify the events and update the corresponding formula templates in the monitors. In order to evaluate and validate the presented approach, the authors set up a comprehensive benchmark with many tests and generated events, based on a simple case study parameterized by the frequency of events and the scale of the involved components.

Planning and monitoring execution with business assertions

The authors of [LAP04] defined monitoring the execution of web services as planning their composition while satisfying the goals stated by the clients in the XSRL language. The goals are expressed by XSRL, using EaGLe language, in the way both express preference among goals and whether they are vital or optional (strong or weak). The functional architecture comprises four components: a planner, a monitor, an executor and a service registry. The monitor holds the central role by looking at client requirements represented as requests in XSRL, expressing their needs in addition to XSAL (XML Service Assertion Language) used by businesses to specify the constraints that must be satisfied when clients use their processes. It then requests the planner for synthesizing a plan that respects the stated goals. If produced, the plan will be passed to the monitor, who in its turn will charge the executor to process it. For invoking suitable services the planner queries the registry and uses the returned information to update its goals and change the current state of the business process. Otherwise, the monitor will relax some constraints, regarding their importance to the goal and requests another plan. The process is repeated until all sub-goals are reached.

Query-based business process monitoring

In [BEM+0707] Beeri et al. propose an approach to the monitoring of business processes specified in BPEL. A visual language, called Business Process Query Language (BPQL), with query capabilities, over BPEL processes, was introduced, for easing the formulation of monitoring queries in the same way that graphical notations help BPEL designers generate specification code, using dedicated icons for each activity. Queries are formulated using the process specification by selecting activities of interest and combining them, using the repetition operator, when some portion of the process is supposed to be reproduced more than once and the alternative operator which is used for selecting between two or more activities. Selecting activities is to be done at different levels of abstraction of the business process specification. A complete process, where all participating actors are taken into account, is defined globally by identifying the activities that are, after that, refined by detailing their sub activities, and so on.

The proposed system by the authors is implemented as follows. A BP-Monitoring query is compiled into a BPEL process specification, whose instances perform the monitoring task, which is translated into an executable code to be run on the same BPEL application server as the monitored business process. An additional component, a so-called dispatcher, is used to listen to the events on the process activities and forward them to the query process instance. An important feature of the approach is that it does not target a particular monitoring goal. Indeed, the reports provide just the required values and may therefore be used for various purposes regarding BPEL processes.

Model-driven development of monitored process

The authors of [MMA07] present a model-driven approach to developing monitored business processes. The authors have created a business performance management metamodel for the modeling of monitoring tasks in a platform-independent way, which allows the modeling of process performance indicators (PPIs) based on BPMN process elements. The BPMN process model with the corresponding PPI model is transformed to a BPEL process model, which contains additional activities for publishing events needed for the calculation of the PPIs. These events are sent to an external monitoring tool by invoking its web service interface. For measuring the duration of the activity, for example, two additional BPEL *invoke* activities would be inserted, before and after the activity, respectively. These activities would invoke corresponding operations on the monitoring tool. The benefit of this kind of approach is that much of the code can be fully generated for the creation of monitoring tool.

Privacy/Security Monitoring

Web services will be affected by dynamic changes of the environment characteristics such as security/privacy constraints. In many cases, application developers and administrators know when adaptive changes would improve system performance. However, they are not able to benefit from it, because the systems usually are not equipped with mechanisms for supporting monitoring and re-configuration.

Monitoring security patterns

In [SKA07, KS07] the authors address the problem of monitoring important security properties of service-based systems. While the static analysis techniques are widely used to check the security properties at design-time, the run-time verification of these properties and the assumptions under which these properties were shown to hold are still required. In order to tackle with this problem, the authors propose to use the techniques described by the previous framework so as to define and monitor basic security properties, namely *confidentiality* (i.e., the absence of unauthorized disclosure of information); *integrity* (i.e., the absence of unauthorized transformations of the state of a system); and *availability* (i.e., the readiness of a system to provide a correct service). These properties are defined using the special patterns modeled as Event Calculus properties, which allows for the monitoring of security properties even to non-expert users.

Traceability-based security monitoring

In [KRL+00] the authors presented dynamicTAO, a CORBA-compliant reflective object-request broker that supports dynamic configuration. It maintains an explicit representation of its own internal structure and uses it to perform runtime customization safety. On top of dynamicTAO, they developed two systems: (1) a flexible monitoring system for distributed objects and (2) a mechanism for enforcing access control based on dynamic security policies.

Auditing Compliance

In [ABF+04], the authors introduced an auditing framework for determining whether a database system is adhering to or compliant with its data disclosure policies. The approach is the following: the user formulates audit queries to specify the data subject to disclosure, and the corresponding audit component returns the queries that tried to access the specified data, during their executions. In [GGG08], the authors extended this approach by designing a framework for pre-computing privacy policy parameters that can assist the auditing officer in formation of a precise audit expression.

Monitoring privacy-agreement compliance

In [BMH07] Benbernou et al. address the problem of run-time monitoring of compliance of the privacy agreement defining the user's privacy rights and their possible handling by the service provider. This problem goes beyond the traditional access control management and defines the necessity to face the usage control management of the private user information.

The proposed solution presents the privacy agreement model, where the requirements on the management and handling of the privacy data are specified, together with the approach for run-time compliance monitoring. The privacy properties are given in the form of data-rights (authorized operations) and data-obligations (required actions) together with their validity frames and specified in the extended WS-Agreement specification. The set of privacy requirements, privacy units, and typical misuse scenarios are defined based on these properties. The formalism adopted for the representation of the privacy units and misuse relies on linear temporal logic. For the monitoring purpose, privacy units are transformed into state machine representation that correspond to the evolution of the privacy data management and define both correct and incorrect usage of these data.

The monitoring framework incorporates three main ingredients, namely: requirements specification, privacy unit observer, and monitor. The requirements specification is transformed into the corresponding privacy unit state machines, which at run-time evolve in parallel with the service execution. The monitor collects the information about the privacy data use from the service logs and updates the status of the privacy unit observer accordingly. The latter reports the violations of the privacy requirements, when a specific failure state of the corresponding state machine is reached.

The proposed framework relies on a clear and simple model of privacy agreements, while the underlying requirements model relies on a comprehensive formalism for the representation of correct usage of private information. The run-time monitoring approach exploits automated techniques for the extraction and execution of monitor programs.

Process Mining

Process mining is an approach of observing and extracting certain knowledge about business processes execution(s) from available event logs [AP07]. In the context of service-based applications these logs may refer to registration of SOAP messages between services, event logs registered by the business process engines, etc.

Fuzzy mining

In [GA07], the problem of simplifying processes on the basis of actual execution traces is proposed. This problem amounts to the reconstruction of a process model with the objective of avoiding the generation of Spaghetti-like models. A Spaghetti-like model [GV07] is defined as a model that is very complex and (usually) useless for humans that have to understand and handle it. These models are neither easily readable (too many elements), nor human-understandable. Therefore, some sort of simplified visualization scheme should be found to present resulting models to users.

The approach proposed in the paper tries to simplify the process schema by preserving (or, at least, trying to degrade as less as possible) two general metrics. *Significance*, defined for both activities and their relations, tries to keep as high as possible the fidelity of the behavior of the model with respect to the behavior of the actual model. *Correlation* is a metric that tries to match the amount of correlation between two activities. Generally, this is measured as the amount of data they share in the recorded events in the log. More precisely, the approach can be sketched as follows:

- Highly significant behavior is preserved, i.e. contained in the simplified model.
- Less significant but highly correlated behavior is aggregated, i.e. clustered together within the simplified model.
- Less significant and lowly correlated behavior is removed from the simplified model.

To estimate significance and correlation to the authors rely on models built upon users' preference. Log-based versions of the metrics are defined to take into account user activities. Derivative metrics, instead, are derived as combination of the previous ones.

Conformance checking with ProM

In [RA08] the authors address the problem of business process conformance checking. Conformance checking deals with the problem of analyzing whether the model (i.e. the workflow on which the business process is modeled) and actual execution traces (registered in the log) conform each other. Conformance analysis aims at the detection of inconsistencies between a process model and an event (i.e. trace) log through appropriate metrics.

entities must not be multiplied beyond what is necessary. This principle became known as Occam's (or Ockham's) Razor or the law of parsimony. A problem should be stated in its basic and simplest terms. In science, the simplest theory that fits the facts of a problem is the one that should be selected.

The authors propose a post-mortem monitoring technique based on two different metrics:

- *Fitness*: does the observed process comply with the control flow specified by the process model?
- *Appropriateness*: does the model describe the observed process in a suitable way? Appropriateness is formulated according to an "Occam's razor" principle[1]: "*entities must not be multiplied beyond what is necessary*". That is, "the simplest theory that fits the facts of a problem is the one that should be selected".

In this work, the model taken into account for processes is Petri Nets and the two metrics above are defined in its terms. Fitness (that is the most dominant requirement for confor-

[1] http://www.2think.org/occams_razor.shtml

mance) is defined as the ability of the Petri Net to “generate” all the traces observed in the log. To be appropriate, a “good” process model should be minimal in structure and minimal in behavior. The paper itself presents a lot of interesting metrics upon which a conformance checking algorithm is built. The conformance analysis in terms of these two metrics may be performed using the ProM toolkit [DMV+05].

A noticeable point of the paper is that the algorithm is very nicely tested on a real-life log of events generated by the use of a town-hall task management application. This is very interesting, actually, since many of the approaches present in literature either give theoretical proof for the validity of their model, or test them on synthetically generated workloads.

4.2.3. Summary

In this section we will summarize information on different contributions and compare different approaches according to the above classification framework. With regards to the *input* dimension, the focus is on languages by considering their specificities. Most of the approaches consider abstractions and then leaving out the implementation details. As for the *task* dimension, the focus is on properties (e.g., temporal properties, security decisions, suspicious queries). The considered aspect is mainly the behavior, either properties directly related to the observation of the execution or those related to the access and exchange (security). For the dimension *processing*, the main issue to be noted is the use of the traces of the executions to get the data on which to perform computation. Also, the techniques vary from a domain to another. Finally, regarding the dimension *Invasiveness*, the idea is to look for a better integration of the proposed framework with the existing environments/architectures without any computation overhead.

To summarize, most of the approaches regarding business processes monitoring and security/privacy monitoring focus on the design of appropriate languages. It follows that many aspects should be captured when dealing with monitoring. This calls for hybrid approaches for monitoring. By hybrid we mean different constraints and specifications that need different semantics and probably different computational models.

Table 6: Summary of the input dimension

		Dynamic monitoring of WS-BPEL processes	**Requirements monitoring**	**Planning and monitoring with business assertions**	**Query-based BP monitoring**	**MDD of monitored process**	**Monitoring security patterns**	**Traceability-based security monitoring**	**Auditing Compliance**	**Monitoring privacy-agreement compliance**	**Fuzzy mining**	**Conformance checking with ProM**
Input	***Type***	Process variables, external variables	Internal/external variables, service operations	Service-operations, process variables	BPEL process events	Process variables	Security parameters	Security decisions	SQL query	Invocation of privacy-critical service operations Invocation of privacy-critical service operations	Process activities	Process activities
	Languages	Special assertion-based notation, WSCoL First-order logic	EC-Assertion language expressed with Event calculus logic	WSAL, Specific planning domain formalism	Special visual control-flow pattern query language using Specific trace-based formalism	UML	Security patterns using Event calculus logic	-	SQL	Privacy extensions to WS-Agreement, Linear temporal logic	Workflow/ Petri nets formalism	Workflow/Petri nets formalism
	Access Mechanism	Pull	Pull	Pull	Pull	Pull	Pull	Pull	Pull	Pull	Pull	Pull

Table 7: Summary of the task dimension

		Dynamic monitoring of ws-bpel processes	Requirements monitoring	Planning and monitoring with business assertions	Query-based BP monitoring	MDD of monitored process	Monitoring security patterns	Traceability-based security monitoring	Auditing Compliance	Monitoring privacy-agreement compliance	Fuzzy mining	Conformance checking with ProM
Task	***Goal***	Analysis (assertions, temporal properties)	Analysis (requirements, contracts)	Planning and adaptation	General, in particular, BAM	BAM	Analysis (security patterns)	Analysis of security decisions	Find query deemed suspicious	Analysis	Analysis	Analysis (conformance)
	Aspect	Behavior	Behavior	Behavior	Various	KPIs	Security	Security	Security	Privacy	Behavior	Behavior
	Output	Identified deviations	Identified deviations	-	KPI	-	-	Adaptation/ Reconfiguration	SQL query	-	-	-

Table 8: Summary of the Processing dimension

		Dynamic monitoring of ws-bpel processes	Requirements monitoring	Planning and monitoring with business assertions	Query-based BP monitoring	MDD of monitored process	Monitoring security patterns	Traceability-based security monitoring	Auditing Compliance	Monitoring privacy-agreement compliance	Fuzzy mining	Conformance checking with ProM
Processing	***Compliance Language***	-	-	-	-	-	-	-	-	-	-	-
	Source	Internal data of the modified process, data that can be obtained by querying external services	Process engine logs	Invocation infrastructure	Events corresponding to BPEL activity instantiation	Internal data of the modified process	Process engine logs	Traces of executions		Service logs	Process logs	Process logs
	Technique	-	First Order Logic reasoning for identifying violations	Specific planning/monitoring algorithms	Specific trace matching algorithm for simulation	Process weaving to introduce monitoring operations	First Order Logic reasoning for identifying violations	Auditing and caching	Triggers and static analysis	Automated extraction of monitors in the form of state-machines	Data mining	Petri net-based analysis, data mining

Table 9: Summary of the Invasiveness dimension

		Dynamic monitoring of ws-bpel processes	**Requirements monitoring**	**Planning and monitoring with business assertions**	**Query-based BP monitoring**	**MDD of monitored process**	**Monitoring security patterns**	**Traceability-based security monitoring**	**Auditing Compliance**	**Monitoring privacy-agreement compliance**	**Fuzzy mining**	**Conformance checking with ProM**
Invasiveness	***Architecture***	The original process code is extended to interact with monitors,	Extension to the process engine	Monitoring is interleaved	Event dispatcher is embedded into the process engine	The original process code is extended to interact with monitors	Extension to the process engine	Integrated with the service execution platform	Integrated with RDBMS	Integrated with the service execution platform	Separated from execution platform	Separated from execution platform
	Execution	Sync (blocking), Async execution	Async	Sync	Async	Sync	Async	-	Async	Async	Async	Async

5. Business intelligence and reporting suites

Leveraging the lessons learned from the previous survey (Section 4), the compliance governance architecture to be developed in WP5 will come with a dashboard for the visualization of analysis and mining results through graphical reports. The development of the dashboard will not be done completely from scratch; as much as possible existing solutions will be reused, especially because the market already offers an interesting set of readily available solutions for the generation of reports.

In this regard, it is worth noting that over the last years we have been witnessing an increasing use of Business Intelligence (BI) solutions, i.e., solutions such as data warehouses, analysis, reporting, and data mining tools. Such tools typically allow IT and business people to query, understand, and analyze their operational and business data in order to make better decisions and, hence, to gain competitive advantage. IT is spreading, and more data is being collected inside companies. Outsourced and networked BI solutions are gaining momentum, and more and more companies (and, hence, source systems) are being involved in the data collection process. In short, BI is being used to analyze more data, more processes, and more sources. In parallel to this trend, from an IT perspective, we have been witnessing a growth of BI and reporting suites on the market.

Keeping in mind the key dashboard features discussed in Section 4.1.3, in the following we discuss a set of BI and reporting solutions, ranging from full-fledged, commercial tools to open-source products, which can be leveraged in the development of the compliance governance dashboard in WP5. Special focus will be paid to open-source solutions.

5.1. Dimensions of analysis

Although assisted by existing reporting suites, the development of effective reporting dashboards is in general a complex task. The design of a dashboard for the visualization of monitoring and analysis results is not limited to the design of the actual reports, comprising the definition of the reports, their appearance and delivery. It also requires taking into account factors like the scheduling of the report generation process, security and access control features, drill down requirements, and end user report customization capabilities. Also, the development of the overall dashboard may benefit from the (possible) integration of the reporting suite with existing ETL tools, which might ease the data integration process.

Based on the previous considerations and on our experience in the development of BI applications we therefore established a set of dimensions in order to assess the main tools available on the market and to highlight strengths and weaknesses. Specifically, we propose the following dimensions: *Report Scheduling*; *Security and Access Control*; *Report Definition Support*; *Report Appearance and Delivery*; *Drill Down Capability*; *Extensibility and Customization*; and *Integration with ETL Tools*. In the following, we present details about each of these dimensions.

Report Scheduling

In many situations, acquiring data and producing certain kinds of reports can be a complex and slow process. In this case, it might be impracticable to perform analysis and report generation on demand. Therefore, the possibility to pre-execute reports in a predefined time and periodicity is an important feature for any BI system.

The *Report Scheduling* dimension therefore highlights whether and how the BI system supports the scheduling and automatic execution of the report generation process.

Security and Access Control

The main goal for a BI system is to provide users with the information they need and to show to the users only the information they are authorized to see. This typically requires security and access control. With the ever growing spread of BI solutions (even in outsources scenarios), lots of efforts are being invested to solve privacy and security issues.

The *Security and Access Control* dimension describes which support for security and access control is provided by a specific BI system and how this feature is implemented.

Report Definition Support

The definition of reports might be a complex task, also requiring domain-specific knowledge. To ease the design of reports, a report editor has the purpose to provide a user friendly interface that allows the user to visually compose reports, to drag and drop objects and charts, and to connect them to data fields previously defined in a dataset.

The *Report Definition Support* dimension analyses whether the BI System assists the users to define their reports and what facilities are available for the report definition phase.

Report Appearance and Delivery

The appearance of the reports is very important; the appearance determines which information is highlighted and how easy it is to interpret its meaning. The possibility to graphically format data helps to make the reports better readable and understandable. Then, there are many ways to access a report, for example by sending the report via mail (with the support to configure the recipients) or giving the possibility to embed the reports into other applications. In this regard, it is also important to know in which format the output is provided, e.g., in common formats like pdf, html, or doc.

The *Report Appearance and Delivery* dimension checks the presentation features provided by the BI system and the available ways to deliver the reports to the users.

Drill down Capability

The possibility to change the analysis perspective and granularity, showing or hiding details and navigating through the data by drilling down into details can help the users to find the right level of abstraction and more easily understand the data.

The *Drill down Capability* dimension verifies whether and how the BI System supports the Drill down Capability.

Report Customizability

Not always are reports developed in the best possible and most effective way. In some cases, end users might for instance need to customize the disposal of the information in the report, to add/modify/remove content, to change visible analysis dimensions or filters in a pivot-table, or to change the type of charts.

The *Report Customizability* dimension highlights how the BI system supports end-user-driven customization of final reports.

Integration with ETL Tools

Data analysis and report generation follows the ETL process. Reporting systems might allow developers to control the ETL processes, sometimes integrating the reporting tool with the ETL tool. This integration might also allow the tracking of a history of the data loads and of lineage information for the data.

The *Integration with ETL Tools* dimension identifies the level of integration between the BI system and the ETL tool (if any).

5.2. Commercial systems

BusinessObjects

BusinessObjects' Intelligence Platform (http://www.businessobjects.com/) is probably the most flexible and scalable business intelligence platform that makes it easy for everyone to discover and share insight for optimal decision-making. Built on SOA, it offers the most extensive set of tools on a single platform and allows IT departments to extend BI to any application or process in any environment. Depending on the size of the business (large, mid-size, and small companies), the platform comes in different flavors, providing effective solutions to each of them. The solution "onDemand" also allows companies to run their BI solutions remotely on a hosted server.

The platform provides a deployment, execution, and management environment for its BI tools, reports, and analytics. The scheduling of automatic report execution is supported by a BusinessObjects module called Crystal Reports Server. The security and Access Control feature is provided, allowing IT managers and system administrators to control user access rights and end user privileges. Creating reports is an easy process by dragging e.g. business indicators onto a page, preformatted templates and wizard-driven interfaces assist in building queries and reports and in conducting analyses in a guided step-by-step process. Results are presented to users by means of a dedicated tool (Information Delivery), guaranteeing interoperability with existing IT infrastructures. The reports are not limited to the information contained within them; users can drill down beyond the scope of the original report, effectively allowing them to access and leverage data coming from individual data marts or the warehouse. The end users can quickly customize reports by dragging and dropping new data elements onto a report also creating custom calculations or data formattings, or by adding additional information to the reports they receive. The BusinessObjects BI module is integrated with BusinessObjects Data Integrator, which supports the ETL process and improves life cycle management.

QlikView

QlikView (http://www.qliktech.com/) is a tool proposed as an alternative to traditional data-warehouse-based systems for BI. It is capable of efficiently storing a large amount of data in main memory by means of a non-relational associative structure called *data cloud*, directly fed by operational data sources. As a result of this design, QlikView removes the need to pre-aggregate data, define complex dimensional hierarchies and generate cubes. QlikView integrates the functions of an environment for developing analysis applications with those of an OLAP interface for accessing and navigating data.

Report scheduling is not featured by QlikView, but it provides a fast Query Engine that loads the data into memory allowing the users to query or sub-set the data instantly to only reveal the data which is relevant to a given user. In addition, QlikView allows users to view the data that is excluded by a selection. Security and Access Control is configured on each document/report by assigning specific access rights to each user, and it is controlled by the Publisher module that is integrated with Microsoft Windows NT4 or Microsoft Active Directory. The report editor provides a rich user interface that makes possible to drag and drop a variety of business elements and to arrange these elements into multiple sheets, including zoom control function, alignment tools, etc. QlikView provides flexible ad-hoc analysis capabilities, powerful analytic applications, and simple printable reports. This allows organizations to deploy QlikView to a variety of different report consumers, e.g., highly skilled analysts doing ad-hoc detailed reporting, executives requiring a dashboard for critical business information, and plant supervisors analyzing output performance. The User Interface (UI) is visually interactive, it offers a multitude of possible chart and table types and varieties thereof; there are list boxes for navigating dimensions; statistic boxes; and many other UI elements. The end user can customize the visualization of the final reports by changing the arrangement of report elements, by querying them through simply clicking on the UI elements, and by changing their visual configuration and filters. Charts, graphs, and tables of all types in QlikView are multidimensional analyses. That is, they show one or more measures (e.g., metrics, KPIs, expressions, etc.) across one or more dimensions (e.g. total sales by region). The major difference is that these calculations are performed as the user clicks (On Demand Calculation Engine) and never prior. QlikView is not integrated with any ETL tool, but it provides internally a script editor to load the data to its own structure.

5.3. Open-source systems

JasperReports/iReport

According to JasperSoft, JasperReports (http://jasperforge.org/sf/projects/jasperreports) is the world's most widely used open-source reporting engine. JasperReports is a Java reporting library that can easily be integrated into whatever (Web) application for generating report, forms, invoices, etc. It provides accelerated report development compared to traditional hand-built approaches, support for any kind of report from dashboards through to print-ready operational reports, high-performance, and massive scalability.

The Report Scheduling and also Security and Access Control features are not implemented in the JasperReports engine, but the report engine can be integrated into other applications that can provide these functions. The iReport is a powerful and easy to use graphical report design tool for JasperReports that simplifies the development of even complex reports through a comprehensive library of chart types, built-in expression builder with syntax checker, object methods list, and wizards, graphical query builders for SQL. JasperReports may be output in PDF, XML, HTML, CSV, XLS, RTF, or TXT, in page-oriented or continuous output style for screen or print. Also advanced features are provided, such as sub-reports for complex layouts and dashboards, conditional printing, multiple data sources of multiple kinds in one report, and internationalization and localization. Also, the reports can be designed with drill down hypertext links. The JasperReports engine is not integrated with an ETL tool, but JasperSoft also provides the JasperETL software that can be used to design and implement an ETL process. The JasperReports engine does not provide end users with customization features for final reports.

BIRT

BIRT (http://www.eclipse.org/birt/phoenix/) is an open-source reporting system that can be integrated with the Java/J2EE applications to produce compelling reports. BIRT provides core reporting features such as report layout, data access and scripting. It has two main components: a report designer based on Eclipse, and a runtime component that can be added to an application server. With BIRT, it is possible to add a rich variety of reports to the applications, such as lists (simplest type of report), charts (to graphically represent numerical data), crosstabs (for the rendering of bi-dimensional data), compound reports, and conventional letters and documents.

The Report Scheduling and also Security and Access Control features, like in JasperReports, are not implemented in BIRT, but the report engine can be integrated in other application that can provide this functions. The BIRT Report Designer has an Eclipse-based set of plug-ins that offers a variety of tools to build reports quickly, giving support to organize the data sources and data sets, providing drag-and-drop creation of the presentation portion of the reports, also a Chart Builder to the chart creation and an Expression Builder for the design of simple scripts. Once the data is ready, BIRT has a wide range of options for presenting them to the user, with tables, charts, text and more. A single data set can appear in multiple ways, and a single report can present data from multiple data sets. Reports present data that are sorted, summarized, filtered and grouped to fit the user's needs. BIRT allows operations such as grouping on sums, percentages of overall totals and more, and also the reports can be designed with drill down hypertext links. The BIRT engine is not integrated with an ETL tool and does not support end user report customization.

SpagoBI

SpagoBI (http://spagobi-info.eng.it/SpagoBISiteENG/target/docs/index.html) is a professional BI suite that is developed and released according to the Free Open Source Software community's practices. It allows the end user to compose the BI platform that best suits his/her needs, also mixing open-source and proprietary products in order to save investments already done, providing first results quickly with a smooth insertion in pre-existing environments. SpagoBI is able to cover all the functional aspects of BI, such as: static and dynamic data organization, inquiring, hidden information discovering by means of the data mining technique, the building of a structured and dynamic publishing and control suite.

SpagoBI is structured into components with the aim of providing for each of the BI functionalities a specific, dedicated module, providing many solutions for each analytical area. These components are responsible for supporting different features, such as: Report Scheduling with the tool Quartz; Security and Access Control with eXo Portal or Liferay Portal; Report Definition Support with Ireport and BIRT; Drill down Capability with Jpivot and Mondrian for OLAP Analysis; Integration with ETL Tool using Talend OpenStudio. The reports are structured like in JasperReports or BIRT adding configuration parameters and also the use of widgets with real-time information. Each report can be run as a portlet and the user can integrate many portlets with information coming from different report engines in containers building reports. The reports can be delivered in many formats and also by e-mail.

Pentaho Open BI Suite

The Pentaho Open BI Suite (http://www.pentaho.com/) provides a full spectrum of BI capabilities including reporting, analysis, dashboards, data mining and data integration. Once Pentaho platform is fully implemented, business gets access to a variety of information, including sales analyses, customers and products profitability, HR reporting, finance analysis and a complex information delivery to the top management.

Pentaho includes an open-source scheduler called Quartz, which can be used to schedule any activity of the system, including running a report. The security features are implemented in

Pentaho BI Platform starting from version 1.6 (current version is 1.7). Report delivery, supported by the JFreeReport engine, can be done via subscriptions to specific reports by setting up a periodical delivery schedule, specifying report parameters and choosing a delivery format. Pentaho Reporting includes also report navigation and report viewer components that can be integrated into portals or web pages. Pentaho provides multiple integrated options for report design, including the Pentaho Report Design Wizard, which is built on top of the Eclipse framework and provides a complete drag-and-drop report design environment. Using Mondrian and Jpivot, Pentaho provides an Analysis module with advanced OLAP functionalities. Pentaho also provides business users with an interactive AJAX-based web interface for self-service report creation. Data integration is realized by an ETL tool called Kettle, providing a graphical user interface for the design of ETL procedures, supporting high scalability and flexibility in data processing.

Others

Besides the above discussed products/open-source solutions, there is also a variety of smaller products or projects that may aid the development of proprietary BI solutions. Representatives are:

- ART (http://art.sourceforge.net/)
- DataVision (http://datavision.sourceforge.net/)
- Open Reports (http://oreports.com/index.php?option=com_frontpage&Itemid=1)
- Jmagallanes (http://jmagallanes.sourceforge.net/en/)
- OpenI (http://openi.sourceforge.net/openi_product.html)
- jCharts (http://jcharts.sourceforge.net/)
- Cewolf (http://cewolf.sourceforge.net/new/index.html)
- JOpenChart (http://sourceforge.net/projects/jopenchart/)
- Chart2D (http://chart2d.sourceforge.net/index.php)
- JChart2d (http://jchart2d.sourceforge.net/)
- JCCKit (http://jcckit.sourceforge.net/index.html)
- JGraphT (http://jgrapht.sourceforge.net/)
- JFreeReport (http://www.jfree.org/jfreereport/index.php)
- KIDS, Key Indicators Display System (http://kids.fao.org/)

5.4. Summary

Table 10 summarizes the above discussion of the reporting and analysis tools. In general, it must be noted that the choice of which reporting engine or tool to adopt in a specific situation – besides depending on hard functional requirements (e.g., the need for drill down capabilities of the reports or user-driven report customization) – also depends on non-functional factors, such as (1) the skills of the user, e.g. the skills of the developer in report design or ETL process development, or of end users that may require (or not) to analyze data at different levels of granularity by drilling down, to customize reports by adding new business elements, or that just want to receive static reports via email; (2) security and access control requirements; (3) the complexity of the project and the data to be analyzed; and (4) the budget of the BI project.

In the case of low budget, the discussed open-source solutions might provide the necessary infrastructure. However, open-source products typically also require higher skills from their users (both developers and end users), as in most cases they do not come as ready integrated platforms. Security and access control mechanisms are adequately supported especially by the discussed commercial tools, while the open-source solutions rely on external tools or applications (indeed, JasperReports, BIRT actually need to be integrated into other applications, in order to be run). Finally, the capability to schedule and automatically generate reports, which could for instance alleviate complexity problems, is generally not well supported.

Table 10 Summary of BI and reporting suites analysis.

	Business-Objects	**QlikView**	**JasperReports/ iReport**	**BIRT**	**SpagoBI**	**Pentaho**
Report Scheduling	Supported by Crystal Reports Server Module	Not supported, but it provides a fast Query Engine that loads data into memory	Supported using external tools	Supported using external tools	Supported by Quartz	Supported by Quartz
Security and Access Control	Supported	Supported	Supported using external tools	Supported using external tools	Supported	Supported
Report Definition Support	Supported	Supported	Supported	Supported	Supported	Supported
Report Appearance and Delivery	Supported by Information delivery tool	Supported by Qlik-View Server	Supported by JasperReports Engine	Supported by BIRT Engine	Supported by JasperReports Engine, Birt Engine, Jpivot and Mondrian	Supported by JFree-Report engine
Drill down Capability	Supported	Supported	Supported using multiple reports and hyperlinks	Supported using multiple reports and hyperlinks	Supported using multiple reports and hyperlinks with JasperReports and BIRT or with pivot tables in Jpivot and Mondrian	Supported by Jpivot and Mondrian
Report Customizability	Supported	Supported	Not supported	Not supported	Supported	Supported
Integration with ETL Tools	Integrated with Business-Objects Data Integrator	It is not integrated with an ETL tool; a script editor loads data into its own structure	Not integrated	Not integrated	Integrated with Talend Open Studio	Integrated with Kettle

6. Outlook: compliance monitoring in COMPAS

In the following, we provide a short overview of which solutions discussed in this deliverable might be used in throughout the development of the COMPAS project. The assumptions and interpretations are based on the best of our knowledge. Please note that the discuss might be subject to variations throughout the project and does not represent any binding agreement among partners.

The goal of the monitoring activity in COMPAS is to assure compliance of business executions in service-oriented environments. As already outlined in Section 4.1.3 monitoring for compliance goes beyond what is currently available in industrial monitoring product available on the market; it is thus not possible to straightforwardly apply one of the monitoring tools discussed in more detail in Section 4, without COMPAS-specific extensions. Such extensions are however not adequately implementable in the discussed products, which means that – leveraging the lessons learned in this deliverable – a COMPAS-specific monitoring solution will be developed as integral part of the overall governance architecture to be developed in WP5. The exact structure of the governance architecture is not yet ready, but in the following we attempt to provide a gross outlook on *very likely* implementation choices.

Regarding the management of compliance-specific events, COMPAS will provide models, languages and architectures with the aim of instrumenting services and process engines so that compliance-related events can be generated, processed and evaluated (in addition to system-level events such as the ones discussed for instance in Section 3 and Section 4). The implementation of processes (service compositions) will be based on process languages such as the Business Process Execution Language (BPEL [ACD+02]), and processes will be executed automatically via a dedicated process engine, e.g. Apache ODE BPEL (http://ode.apache.org/ws-bpel-20.html). For the generation of compliance events, both the process language and the engine's possible event model (e.g. BPEL and BPEL's current event model [KKS+06]) will be extended with COMPAS-specific constructs able to express higher-level concerns directly related to compliance concerns. For the monitoring of process executions this means that either (i) runtime data is *pulled* out of the process log/database, or (ii) proper execution events are *pushed* from the engine to the monitoring system. We expect the second of these options will be used, preferably in combination with an external event processing engine such as, for instance, Esper (http://esper.codehaus.org/), a publish/subscribe solution for event processing. Similarly to what the monitoring products discussed in Section 4 do, events will be collected in a central data warehouse for further analysis, data mining and reporting. For the implementation of the governance dashboard that will publish the compliance analysis and mining results to the final user, preference will be given to the open-source solutions (e.g. JasperReports or BIRT), discussed in Section 5. The possibility to integrate such solutions into other applications makes them indeed particularly suited for the development of web-based dashboards. Hence, in line with the web-based interpretation (services and SOA) of the business execution, also the compliance governance instruments developed in the context of COMPAS will come with a web interface for the end user.

7. Conclusion

Service and service composition monitoring are the first step toward what we call compliance monitoring, which, in turn, is one of the ingredients for a comprehensive compliance governance architecture. In this deliverable, we have investigated several aspects of service monitoring, i.e., (i) we have looked at *current practices* in industrial products in order to understand what features users expect from their monitoring applications and what kind of information is typically published in the final dashboard; (ii) we have then looked at state-of-the-art *research approaches* related to the monitoring problem, with a special focus on compliance-oriented approaches, in order to better understand current solutions and algorithms; and (iii) finally we have looked at business intelligence and *reporting suites* in order to understand how the visualization dashboard can be implemented.

After this analysis of the state of the art in monitoring practices, we have then provided an outlook on how we believe the discussed products and approaches (or a selection thereof) will work together and fit into one integrated platform for compliance governance.

From this survey, it follows that many aspects should be captured when dealing with monitoring. This calls for hybrid approaches for monitoring. By hybrid we mean different constraints and specifications that need different semantics and probably different computational models.

8. Reference documents

8.1. Internal documents

[D2.1] "State-of-the-art in the field of compliance languages", version 0.99 of 2008-07-21

[D8.1] "Project Quality Plan", version 1.00 of 2008-04-30

[DoW] "Description of Work" for COMPAS, final version of 2008-02-01

8.2. External documents

[ABF+04] Rakesh Agrawal, Roberto Bayardo, Christos Faloutsos, Jerry Kiernan, Ralf Rantzau and Ramakrishnan Srikant. Auditing compliance with a Hippocratic database. Proceedings of the Thirtieth international conference on Very large data bases (VLDB 2004). Toronto, Canada. Pages 516—527, 2004.

[ACD+02] T. Andrews, F. Curbera, H. Dholakia, Y. Goland, J. Klein, F. Leymann, K. Liu, D. Roller, D. Smith, S. Thatte, I. Trickovic, S. Weerawarana. Business Process Execution Language for Web Services version 1.1. July 2002. http://www.ibm.com/developerworks/library/specification/ws-bpel/

[Barlas06] Demir Barlas, Line56.com. "Monster Integration", October 25, 2006, http://www.oracle.com/technology/products/integration/bam/pdf/BAMREFERENCE_MONSTERCOM.pdf

[BBG+07] L. Baresi, D. Bianculli, C. Ghezzi, S. Guinea, P. Spoletini, "A Timed Extension of WSCoL" in Service-Oriented Computing - ICSOC 2007, Fifth International Conference, 2007, pp. 663–670.

[BEM+07] Catriel Beeri, Anat Eyal, Tova Milo and Alon Pilberg. Monitoring Business Processes with Queries. VLDB 2007. Pages 603-614.

[BG05] L. Baresi and S. Guinea, "Towards dynamic monitoring of ws-bpel processes," in Service-Oriented Computing - ICSOC 2005, Third International Conference, 2005, pp. 269–282.

[BJB+07] D. Bianculli, R. Jurca, W. Binder, C. Ghezzi, and B. Faltings, “Automated dynamic maintenance of composite services based on service reputation,” in Service-Oriented Computing - ICSOC 2007, Fifth International Conference, 2007.

[BMH07] S. Benbernou, H. Meziane, and M.S. Hacid, “Run-Time Monitoring for Privacy-Agreement Compliance”, in Service-Oriented Computing - ICSOC 2007, Fifth International Conference, 2007, 353 – 364.

[CA07] CA. “Unicenter Service Assure r11.1”, 2007, http://ca.com/Files/Product Briefs/service_assure_product_brief.pdf

[DGR04] N. Delgado, A. Q. Gates, and S. Roach, “A taxonomy and catalog of run-time software-fault monitoring tools,” IEEE Trans. Softw. Eng., vol. 30, no. 12, pp. 859–872, 2004.

[DMV+05] B.F. van Dongen, A.K. A. de Medeiros, H.M.W. Verbeek, A.J.M.M. Weijters, and W. M. P. van der Aalst, “The ProM Framework: a New Era in Process Mining Tool Support”, in Applications and Theory of Petri Nets 2005, pp. 444 – 454.

[GA07] C. W. Günther, W.M.P. van der Aalst, “Fuzzy Mining - Adaptive Process Simplification Based on Multi-perspective Metrics”, in BPM 2007, pp. 328-343.

[GG07] C. Ghezzi and S. Guinea, “Run-time monitoring in service-oriented architectures,” in Test and Analysis of Web Services, L. Baresi and E. D. Nitto, Eds. Springer, 2007, pp. 237–264.

[GGG08] S. K. Gupta, Vikram Goyal and Anand Gupta. Precomputation of privacy policy parameters for auditing SQL queries. Proceedings of the 2nd international conference on Ubiquitous information management and communication. Suwon, Korea. Pages 87-93. 2008.

[GV07] C. W. Günther and W.M.P. van der Aalst. Fuzzy Mining: Adaptive Process Simplification Based on Multi-perspective Metrics. In G. Alonso, P. Dadam, and M. Rosemann, editors, International Conference on Business Process Management (BPM 2007), volume 4714 of Lecture Notes in Computer Science, pages 328-343. Springer-Verlag, Berlin, 2007.

[HP00] Hewlett-Packard Development Company, L.P. “HP Stakes Claim in Mobile e-Services Management Market”, Press release of Nov. 14, 2000, http://www.hp.com/hpinfo/newsroom/press/2000/001114a.html?jumpid=reg_R1002_USEN

[HP03] Hewlett-Packard Development Company, L.P. “Con Edison Communications Turns to HP to Enhance Customer Service Through IT Infrastructure Consolidation”, Press release of Oct. 27, 2003, http://www.hp.com/hpinfo/newsroom/press/2003/031027a.html?jumpid=reg_R1002_USEN

[KKS+06] D. Karastoyanova, R. Khalaf, R. Schroth, M. Paluszek, F. Leymann. BPEL Event Model. Technical Report 2006/10, November 2006, Institute of Architecture of Application Systems (IAAS), Stuttgart.

[KRL+00] F. Kon, M. Roman, P. Liu, J. Mao, T. Yamane, L. C. Magalhes, R. H. Campbell, "Monitoring, Security, and Dynamic Configuration with the dynamicTAO Reflective ORB", To appear in Proceedings of the IFIP International Conference on Distributed Systems Platform and Open Distributed Processing (Middleware2000).

[KS07] C. Kloukinas and G. Spanoudakis, “A Pattern-driven Framework for Monitoring Security and Dependability”, in TRUSTBUS’07.

[LAP04] A. Lazovik, M. Aiello, and M. Papazoglou, “Associating Assertions with Business Processes and Monitoring their Execution”, in Service-Oriented Computing - ICSOC 2004, Second International Conference.

[McCoy02] D.W. McCoy. Business Activity Monitoring: Calm Before the Storm. Gartner Research, ID Number LE-15-9727, April 2002. http://www.gartner.com/resources/105500/105562/105562.pdf.

[MMA07] C. Momm, R. Malec, S.Abeck, “Towards a Model-driven Development of Monitored Processes”, Wirtschaftsinformatik (2) 2007: 319-336.

[MS05] K. Mahbub and G. Spanoudakis, “Run-Time Monitoring of Requirements for Systems Composed of Web-Services: Initial Implementation and Evaluation Experience”, in 2005 IEEE International Conference on Web Services (ICWS’2005), pp. 257-265.

[Oracle06a] Oracle. “METRO Group Pioneers RFID, Reducing Losses Arising from Inaccurate Stock Replenishment in Retail Stores”, Oracle Customer Snapshop, http://www.oracle.com/customers/snapshots/metrogroup-snapshot.pdf

[Oracle06b] Oracle. “Standards-Based Technology Positions AR Telecom for Future Growth”, Oracle Customer Snapshop,

[PT07] M. Pistore and P. Traverso, “Assumption-based composition and monitoring of web services,” in Test and Analysis of Web Services, L. Baresi and E. D. Nitto, Eds. Springer, 2007, pp. 307–335.

[Ptak07] Ptak, Noel & Associates LLC. “ADP leverages IBM business service management (BSM) solutions to keep corporate payrolls humming”, 2007, ftp://ftp.software.ibm.com/software/tivoli/casestudies/PNA_IBM_ADP_BSM_casestudy_final.pdf/

[RA08] A. Rozinat, W. M. P. van der Aalst, “Conformance Checking of Processes Based on Monitoring Real Behavior”, Inf. Syst. (IS) 33(1), pp. 64-95 (2008).

[SKA07] G. Spanoudakis, C. Kloukinas, and K. Androutsopoulos, “Towards Security Monitoring Patterns”, in SAC’2007.