

D5.6

Version: 1.0

Date: 2010-12-30

Dissemination status: PU

Document reference: D5.6



Final Prototype of Compliance Mining Tool

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 Technische Universitaet Wien (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 Telcordia Poland (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

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

D5.6 Compliance Mining Tool

Revision 1.0

Due date of deliverable: 2010-12-31

Actual submission date: 2010-12-30

Organisation name of lead partner for this deliverable:

UCBL – Université Claude Bernard Lyon 1, France

Contributing partner(s):

PWC – Price waterhouse coopers Accountants N.V., Netherlands

TARC-PL – Telcordia, Poland

THALES – Thales Services SAS, France

TUV – Technische Universitaet Wien, Austria

UNITN – University of Trento, Italy

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	2010-11-15	All sections	New document	UCBL
0.2	2010-12-15	All sections	Updates	UCBL
0.3	2010-12-25	All sections	Create an enriched version of the document	UCBL
0.4	2010-12-26	Specific sections	Update of the document according to the review from UNITN	UCBL
1.0	2010-12-30		Approval	TUV

Authorisation

No.	Action	Company/Name	Date
1	Prepared	UCBL	2010-12-26
2	Approved	TUV	2010-12-30
3	Released	TUV	2010-12-30

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.

The document is proprietary of the COMPAS consortium members. No copying or distributing, in any form or by any means, is allowed without the prior written agreement of the owner of the property rights.

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

1. Introduction	6
1.1. Purpose and scope	6
1.2. Document overview	7
1.3. Abbreviations and acronyms	7
2. User guide for the CMT prototype	8
2.1. Compliance Mining tool.....	8
2.2. <i>Dynamic Model Behaviour Simulator</i>	11
3. Implementation details	13
4. Source Code and additional information.....	16
5. Conclusion.....	16
6. Reference documents	16
6.1. Internal documents	17
6.2. External documents	17

List of figures

Figure 1 Compliance governance runtime architecture, an extension of [D5.3]	7
Figure 2	9
Figure 3	9
Figure 4	10
Figure 5	11
Figure 6	12
Figure 7 DOBS model of the WatchMe scenario	13
Figure 9 Main interface of DOBS	15
Figure 10 Correlation of two generated events of the WatchMe use-case scenario	15

Abstract

Compliance governance relies on the fundamental endeavour of mining business process models and Web service business protocols from log files. Model mining aims at the discovery of the behaviour of a running model implementation using solely its interaction and activity traces, and no information on the target model.

In order to support compliance governance we make use of the *Compliance Mining Tool* (CMT), whose design and implementation is an important challenge for at least three reasons: (i) a minority of interaction data is recorded by process and service-aware architectures, (ii) a limited number of methods achieve model extraction without knowledge of either positive process and protocol instances or the information to infer them, and (iii) the existing approaches rely on restrictive assumptions that only a fraction of real-world Web services satisfy.

Enabling the extraction of these interaction models from activity logs based on realistic hypothesis necessitates: (i) approaches that make abstraction of the business context in order to allow their extended and generic usage, and (ii) tools for assessing the mining result through implementation of the process and service life-cycle. Moreover, since interaction logs are often incomplete, uncertain and contain errors, mining approaches proposed in this work need to be capable of handling these imperfections properly.

We propose a set of mathematical models that encompass the different aspects of process and protocol mining. The extraction approaches that we present, issued from linear algebra, allow us to extract the business protocol while merging the classic model mining stages. On the other hand, our protocol representation based on time series of *flow density variations* makes it possible to recover the temporal order of execution of events and messages in the process. In the end, we present a multitask framework aimed at supporting all the steps of the process workflow and business protocol life-cycle from design to optimization.

All these issues are detailed in *Message Correlation and Web Service Protocol Mining from Inaccurate Logs* published at the ICWS'10 conference [MYD+10], in *DOBS: A Dynamic Model Behaviour Simulator for Model Assessment and Mining*, currently submitted at the Sigmod Records journal [MDH+10], and in *Timed transition discovery from web service conversation logs* [DMD+08]. These references provide the required background on the context, issues and solutions of the problem.

This deliverable limits its objectives in providing the information on the location of sources, instructions and implementation information on the compliance mining tool (CMT).

1. Introduction

Compliance is a term generally used to refer to the conformance to a set of laws, regulations, contracts, or best practices (compliance sources according to COMPAS conceptual model [D7.1]).

Compliance mining refers to the usage of mining and model assessment techniques in order to obtain the behavioural model of a business process or a Web service protocol from their event logs [D7.1].

Model assessment (also referred to as model compliance) addresses the issue of conformance towards a policy, standard, law or technical specification that has been clearly defined. This also includes but is not limited to conform towards business regulations and stated user service requirements [D7.1]. As the authors in [EY09] point out in their survey, compliance toward regulations is finding more and more attention in the eyes of the research community. Applications of model assessment include workflow checking, protocol verification, and constraint validation [MMZ05]. Factors that motivate model compliance utility are: cost of the implementation prototype just for assessing the model, cost of re-implementing once that assessment results are negative, risk of testing on real-world already deployed systems, complexity of designed systems which prohibits exhaustive static verification and validation.

Concrete and oriented applications are mining of workflows, business processes and software specifications, business protocol discovery and web application behaviour extraction. Applications include: post-mortem monitoring, checking the equivalence between the specification and implementation, obtaining the specification if it does not exist, checking for security flaws, verifying that constraints in the execution flows are satisfied, checking if the designed model is correct, complete and finite (i.e. no deadlocks, infinite loops, bottlenecks).

For these, and many other reasons it is of outmost importance to answer questions such as: *Does the behaviour of the implemented service or process match the design specification model? If not, in which parts do the two models diverge? How does this affect the implemented services and processes regarding their corresponding compliance requirements?*

To address these and similar compliance problems, COMPAS proposes a conceptual model for compliance [D7.1] and for the CMT, as well as the implemented prototype. The aim of CMT is to provide the behavioural model of the services and processes that generate events, in order to allow the expert to study the result of the mined model in the *Compliance Governance Dashboard (CGD)* [D5.5]. This analysis will result in a comparison with the model of the desired behaviour in order to detect possible problems or violations, and to facilitate the identification of root causes for non-compliant situations.

In that context, this deliverable describes how to download and deploy the CMT and provides additional information about the location of details on the design and development of the CMT.

1.1. Purpose and scope

Figure 1 provides an illustration of the relationship between CMT and the other components of the COMPAS compliance governance runtime architecture. This figure is a partial view of the overall architecture described in [DA.1].

From Figure 1 we observe that all the events issued from all the sources (i.e., Business process engine, Complex Event Processing, and Business protocol monitoring) are published

in the ESB (Enterprise Service Bus), in order to be stored in the Event log. This Event log represents the unique and direct input of CMT. After applying the mining algorithms on the events contained in this log, the output of CMT will be provided as a simulation model, designed using the Dynamic Model Behavioural Simulator (DOBS) graphical interface.

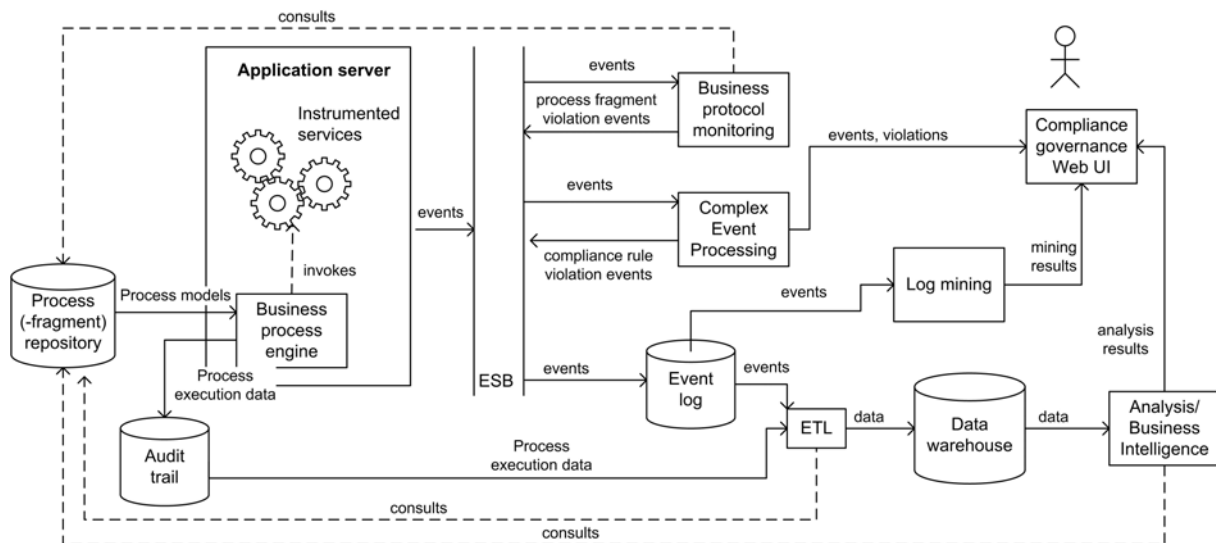


Figure 1 Compliance governance runtime architecture, an extension of [D5.3]

The Complex event processing and Business protocol monitoring, which result in runtime monitoring, were respectively provided by Telcordia and UBCL in the [D5.4]. The Compliance governance dashboard was developed by UNITN and its results are documented in the [D5.5].

1.2. Document overview

The deliverable is organized as follows: Section 2 provides a user guide for the CMT prototype. Section 3 gives an overview on the implementation details of the tools described in this document. Section 4 provides information on where the reader can download the source code of the CMT and find additional documentation and multimedia sources related to the CMT. Finally, Section 5 draws conclusions on the CMT prototype.

1.3. Abbreviations and acronyms

CGD	Compliance Governance Dashboard
GUI	Graphical User Interface
DoW	Description of Work
DW	Data Warehouse
IT	Information Technology
DOBS	Dynamic mOdel Behaviour Simulator
SC	Simulator Controller
SOA	Service-Oriented Architecture

2. User guide for the CMT prototype

The current version of the CMT utilises only two attributes of the events stored in the Event log. These are the event label (or name), and the event timestamp. Both of these attributes are guaranteed to exist in the Event log, according to the [D5.3].

2.1. Compliance Mining tool

The CMT consists of two separate sets of functions (Delta and Gamma) that, combined together, allow the extraction of the behavioural model. A third part of the CMT, i.e. DOBS, is necessary for graphically modelling the execution protocol, and for simulation and testing purposes.

The CMT as a whole is intended to be run on the Matlab program [MM10]. More specifically, the Delta and Gamma extraction libraries are to be called from the Matlab command-line, and the DOBS tool runs on the Simulink platform included in Matlab.

- *The Delta method*

The process of mining the behavioural model starts with the Delta method which provides a first version of the model in the form of a non-oriented graph. The main components of the library of this method are:

- **correlate.m**: the Matlab source file of the implementation of the Delta method. To use this file the user will need to load the file containing the occurrence log in Matlab, and then type on the command line ">correlate(A)". The correlation matrix will be output on the Matlab console.
- **chrono.m**: a simple Matlab source file that allows to measure the execution time of the Delta algorithm. To use it, the user will simply need to type ">chrono(A)" after having loaded the corresponding occurrence log "A" in the workspace.

The occurrence log records the number of occurrences for each type of event encountered in the initial Event log. An example illustrating the view of an occurrence log is given in Figure 2.

The Delta method allows obtaining the matrix of coefficients from the initial occurrence log. These coefficients will be employed in order to reconstruct the states of the model, associated with the events related to a particular state. An example of how the user should proceed in order to obtain such a result will be provided in the following. An illustration of the matrix of coefficients is provided in Figure 3.

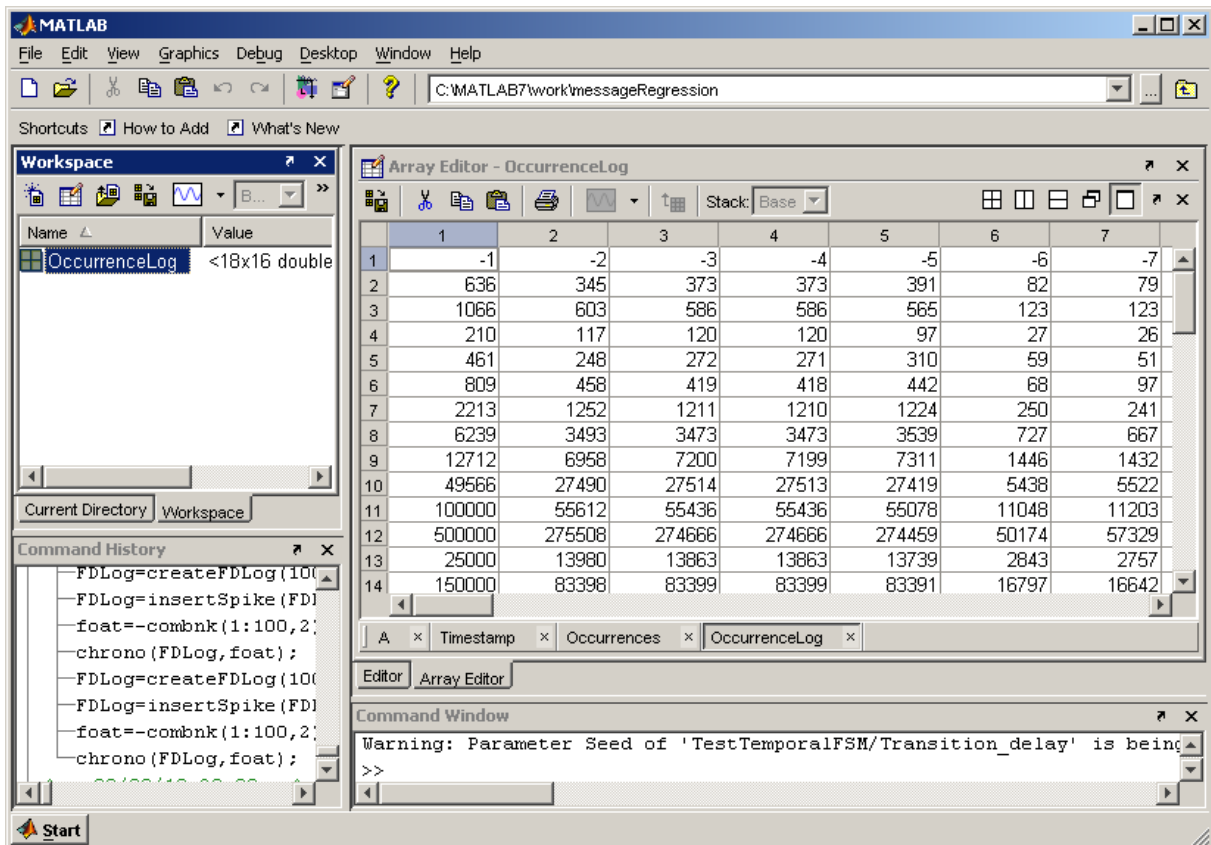


Figure 2 Occurrence log loaded onto the workspace

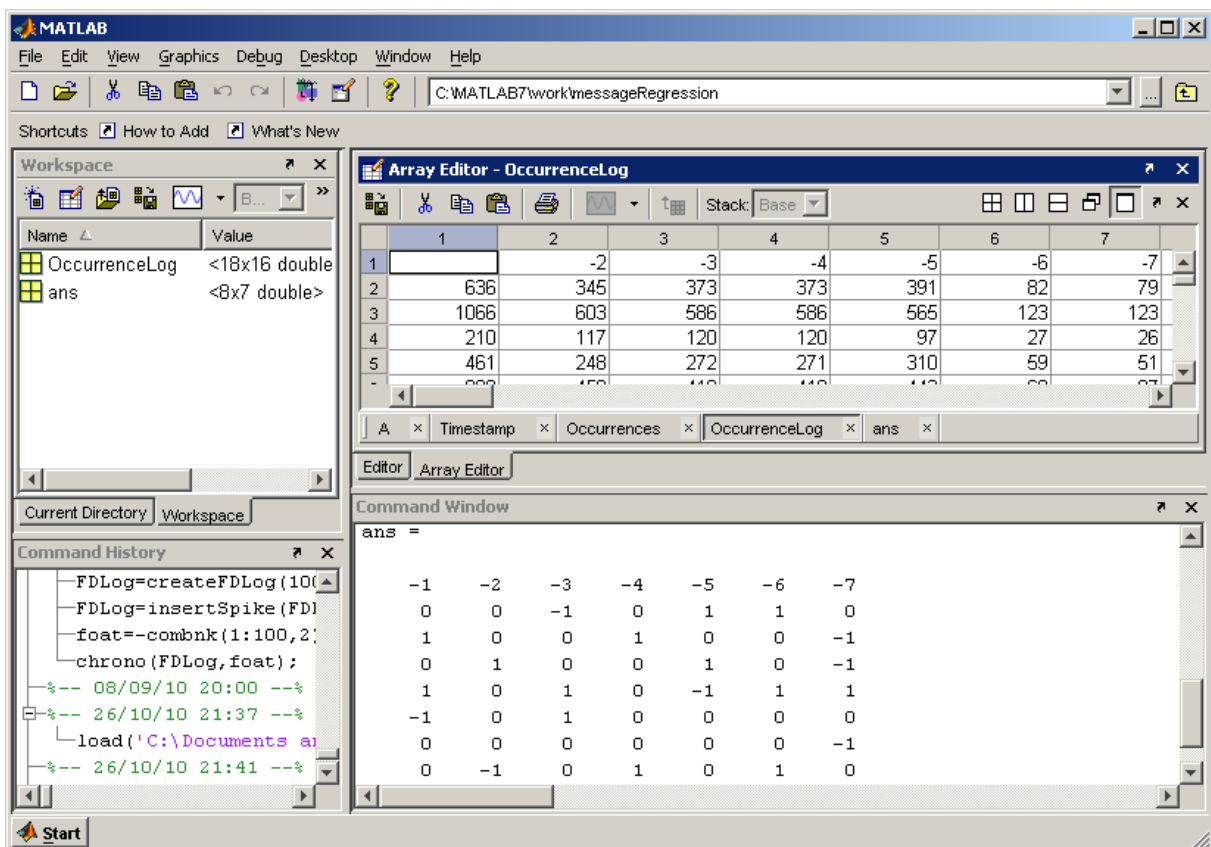


Figure 3 Matrix of coefficients obtained from Delta

- *The Gamma method*

The process of mining the behavioural model continues afterwards with the Gamma method, which is employed for obtaining the mined information that allows to obtain the temporal graph of the model, and thus all the necessary information for designing the final behavioural model using the DOBS GUI. The main components of the library of this method are:

- **createPlotData.m**: the function that allows computing the flow density log starting from the Event log in Figure 1. The flow density log stores the number of occurrences of events whose timestamps are included in the same time interval (leftmost column of the example matrix); an illustration is shown in Figure 4.
- **gamma.m** and **checkAffineTransformation.m**: the Matlab source files of the implementation of the Gamma method.
- **temporalOrder.m**: the Matlab source file that the end-user will execute, once the Delta approach will have yielded the coefficients matrix, and the main function that the end-user will need to call from the command line in order to utilise the Gamma method.

The reader should be aware that in order to use this implementation of the Gamma algorithm at full capacity, it will be necessary to run the Delta algorithm on an occurrence log first, and then feed the resulting array of coefficients built from the output of Delta, as input to "chrono.m" or "temporalOrder.m".

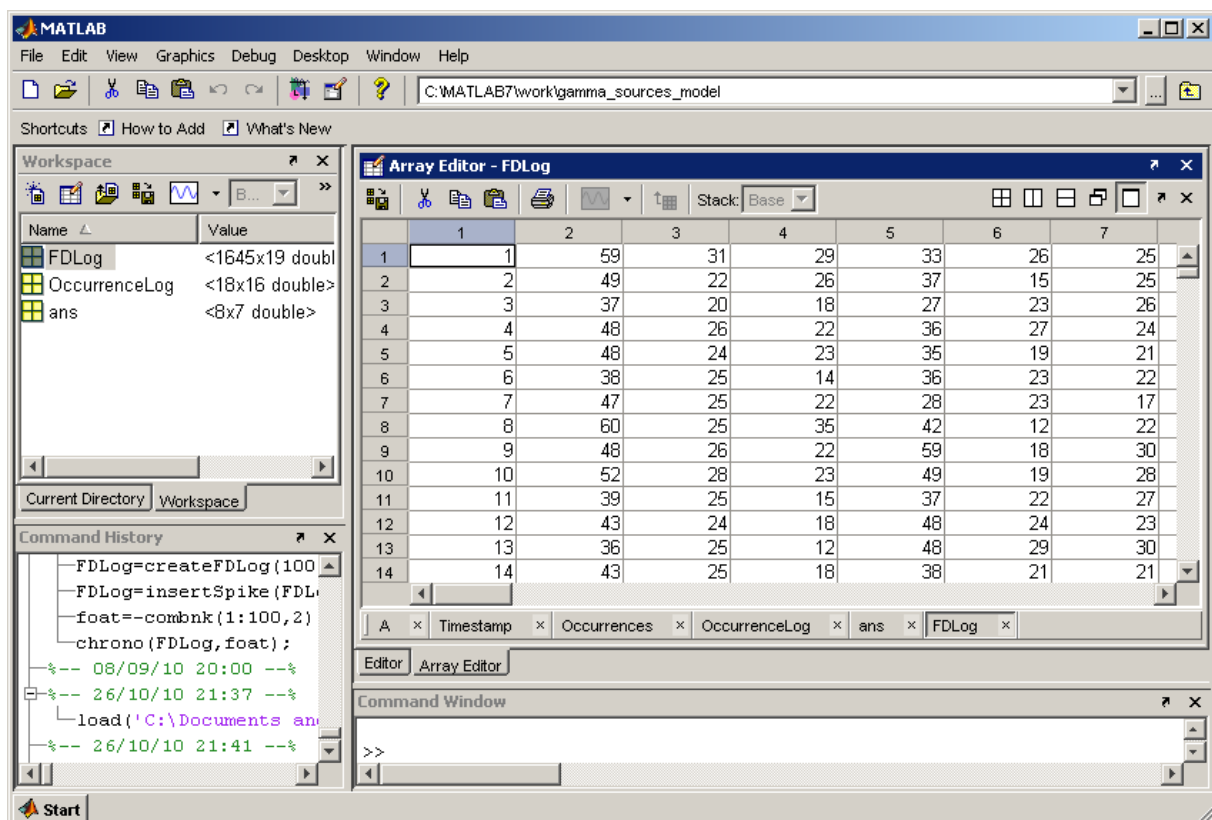


Figure 4 Flow density log obtained from an Event log

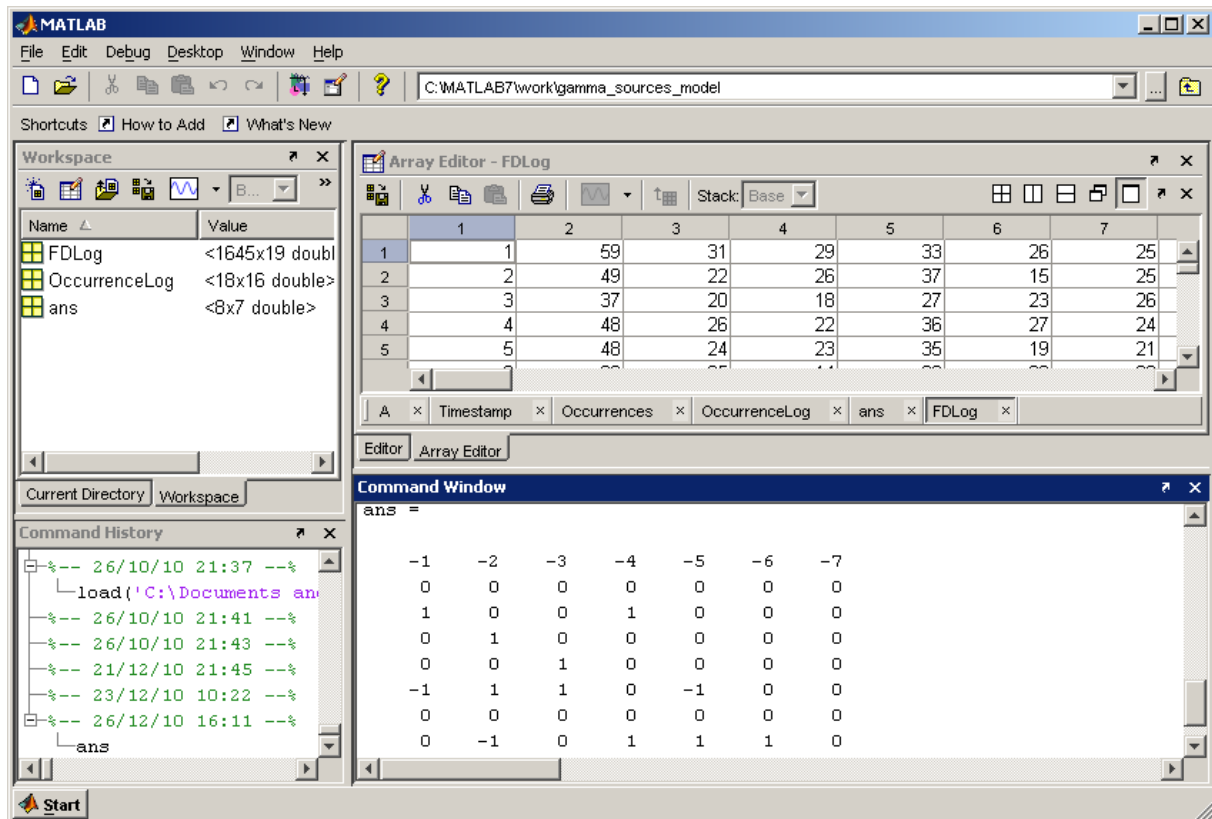


Figure 5 The temporal matrix computed from Gamma based on a flow density log

The temporal matrix, an example of which is given in Figure 5, provides the last information on the temporal order between events, and opens the path to the design of the visual model of the final extracted protocol. In this example, the second event type symbolized by the second column of the array occurs *before* (code value +1) the event designated by the third column of the array, and *after* (code value -1) the event designated by the seventh column of the array.

2.2. Dynamic Model Behaviour Simulator

DOBS is an analysis and decision support compliance verifier and simulation tool for data generation, optimization and validation of compliance in processes and services. The core of DOBS is a discrete event simulator that allows for the reproduction of the behaviour of dynamic models. DOBS is thus mainly designed for simulating business processes, web services, workflows and software control charts. This can be used to play what-if scenarios in case of a future update, allowing experts to assess problematic points, and the impact of every modification on the evaluation criteria. By doing so, DOBS avoids the expensive and time-consuming task of implementing changes on the software/real application level.

The main objectives of DOBS are to assess and comply with (i) the quality of the model behaviour in terms of completeness, scalability, and consistency, and (ii) the properties and quality of generated data in terms of temporal consistency, pattern coherence, and statistical properties. DOBS also targets the generation of text logs from live executions of processes and service.

The DOBS tool serves as the graphical user interface that will serve, amongst other purposes, as the output display of the model mining process. The design of the extracted model requires inevitably the intervention of the end-user, since the interpretation and transformation of the

results of Delta and Gamma requires an analysis that cannot be fully automatic for semantic reasons.

The matrix of coefficients provided by Delta is to be interpreted as follows: we consider each column at a time. If for row k and column l the value i_{kl} is 0, then the message type corresponding to that row is not correlated to the message type associated with column l . If i_{kl} is ± 1 , then the two event types are associated to a given state. Thus, each column expresses via a proper linear equation the correlation between the corresponding event type and the other types of events. More specifically, for each column of the array of coefficients (for example the one in Figure 3), a state is built and the transitions are added based on the signs of their corresponding coefficients. Figure 6 illustrates the construction of three particular cases. This example also shows why the user intervention is mandatory during the design process. Indeed, only a user is capable of recognizing that Figure 6 c) is in fact the same as if the states in Figure 6 a) and b), were put in a sequence using an additional state between the two of them.

The direction of the transitions is determined using the temporal matrix provided by Gamma. The interpretation of the values in this matrix has already been given in the preceding subsection.

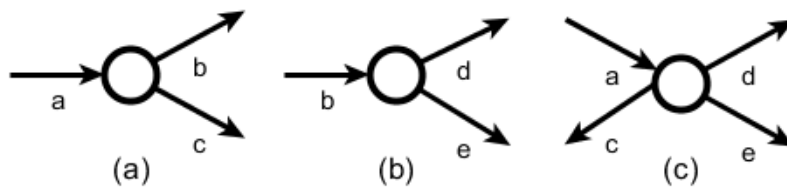


Figure 6 Building the states of the model

The model design takes part in what is referred to as the *simulator controller*. An example of this interface is given in Figure 7. The main elements composing the diagram in this interface are the states, and the transitions connecting them. In this example, one observes additional information that can be specified during the design phase, in order to configure the extracted model for further simulation and testing. This information can be divided as follows:

- The **Input Data** module receives all incoming data from outside the SC and eventually prepares them for further usage.
- The **Elementary Modules**, which will compose the automata, whose execution represents the behaviour of the model. These modules include states, transitions, super-states, junction points, user-written functions, etc.
- The **Internal Logic Statements** is composed of single and optional statements such as variable instantiation, arithmetic operations and so on. These statements are edited and inserted directly into the transition labels of the automata, and executed if and only if the event identifying the transition is triggered and the associated condition returns the Boolean *true* value.

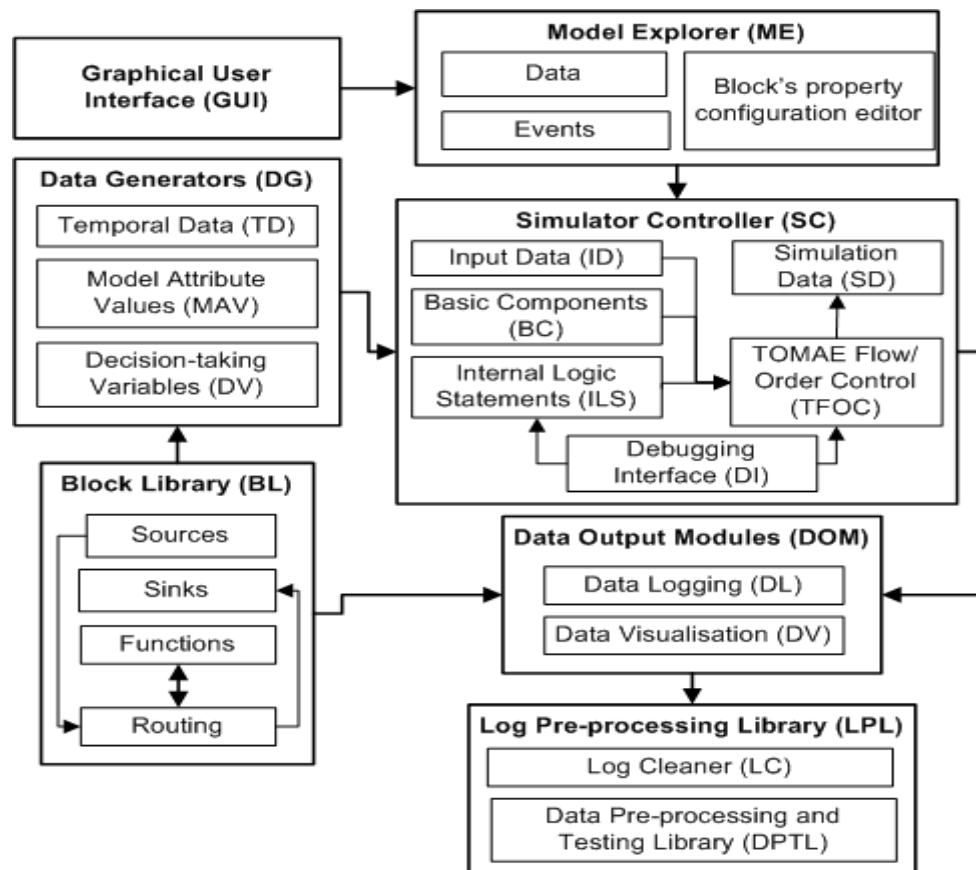


Figure 8 Conceptual model of DOBS

- The **Block Library Module** provides all the elementary blocks that will compose every model.
- The **Log Pre-processing Library** constitutes the final stage of the DOBS usage flow, and deals with the crucial task of (i) cleaning logs from redundant and other irrelevant data, and (ii) reorganizing data structures and manipulating data content in order to make it fit for further usage, as well as testing its properties in order to ensure that the output corresponds to the users' expectations before feeding the output to furthermore processing and analysis steps.

Figure 9 provides the view of the main graphical interface that encompasses all the components of DOBS in its entirety. It is important to notice the grey module in the center, standing for the simulation controller, and containing the mined model that is designed based on the results of Delta and Gamma.

Figure 10 illustrates the correlation properties of two event types, *LoginSuccess*, and *SearchMedia*, which were generated from the post-mining design model on DOBS of the WatchMe use-case scenario. This example also provides a glimpse on the visualisation capabilities of the execution environment.

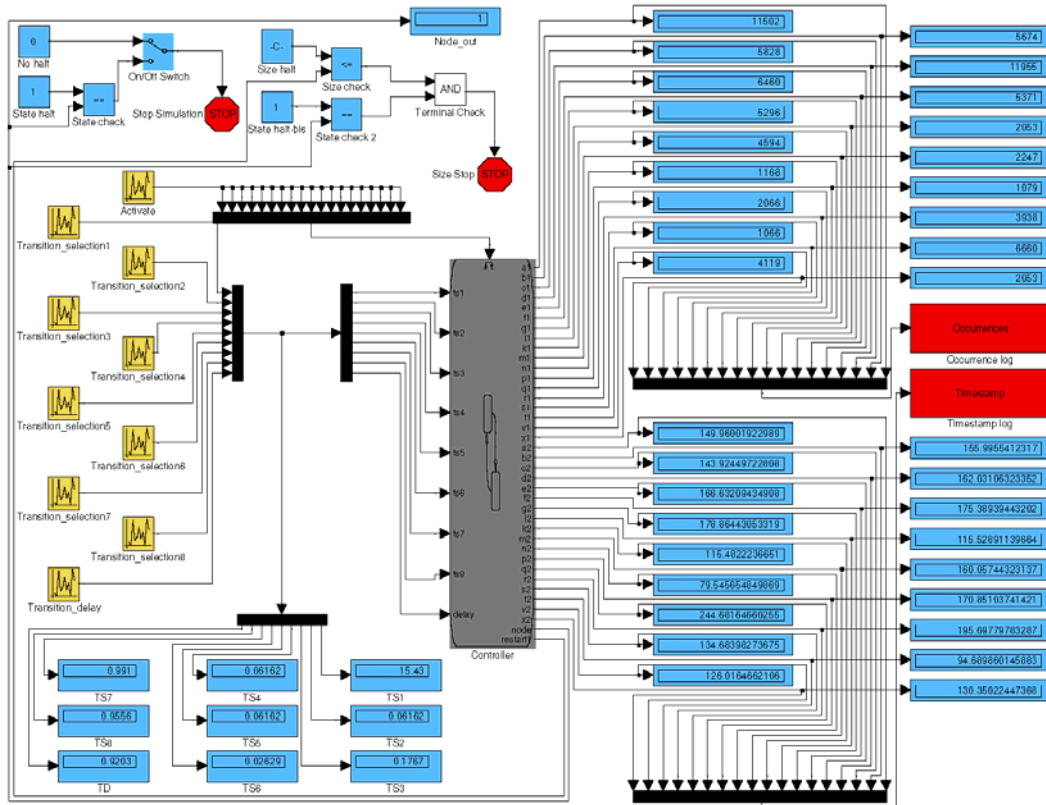


Figure 9 Main interface of DOBS

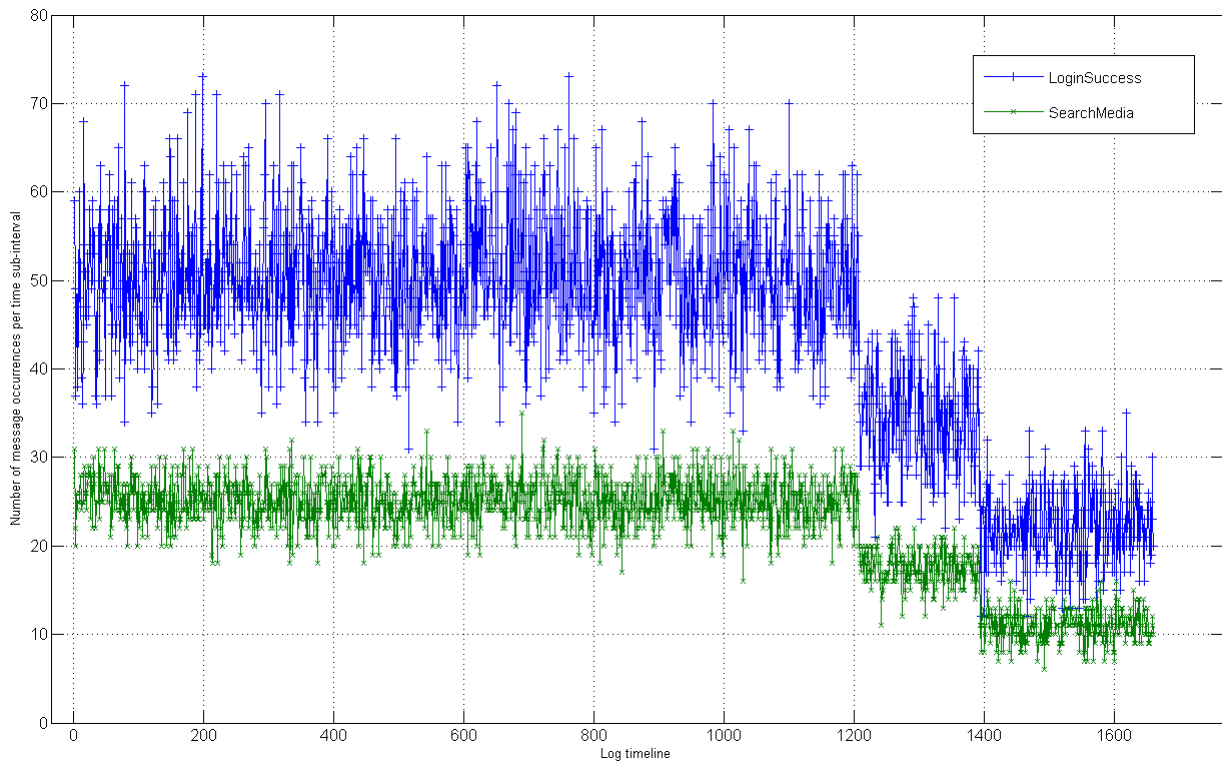


Figure 10 Correlation of two generated events of the WatchMe use-case scenario

4. Source Code and additional information

The CMT code is released as an open source project under the GNU General Public License (GPLv3). Source Forge [SF10] was employed as the tool of choice for disseminating the prototype onto the general public.

The sources are available at <https://cminingtools.svn.sourceforge.net/svnroot/cminingtools/> using the following credentials login: cmtcompas and password: compas. The code is organized in two separate folders: the “CMT” folder contains the code of the two sets of mining algorithms and the “models” folder contains some sample models for the simulation and generation of data. Each zip archive contains detailed instructions on how to employ the models and algorithms.

The CMT prototype can be deployed locally on any Matlab platform and on a network if the user has the appropriate licensing. Currently, CMT has been tested on sample data in order to extract the behavioural models of two case study scenarios explored by the COMPAS project, named Loan Approval (Banking CGD) and WatchMe (Telecom CGD).

A video illustrating the GUI of DOBS, and further details on CMT, and the models used can be found here: http://liris.cnrs.fr/~kmusaraj/files/technology/simulation/DOBS_demo.avi.

5. Conclusion

In this deliverable we introduced the Compliance Mining tool, whose objective is to extract the behavioural model of services and processes from their event logs. The main usage of such a model is to analyze the potential discrepancies existing between the design-time specification and the real-world implementation. We went through the mining methods, namely Delta and Gamma, and introduced their operating requirements and instructions.

Furthermore, we indicated how to design graphically the mining results using DOBS, in order to provide not only the possibility of displaying a graphical representation of the mined model, but also to simulate this model and test its what-if capabilities. In the end, we indicated the location of the libraries, demos and documentation on the components of the CMT. In the end, we would like to point out that the future evolution of the CMT remains an open issue and a promising perspective.

6. Reference documents

As mentioned in the abstract, the approaches and implementation of the CMT are described in the following documents:

- K. Musaraj, T. Yoshida, F. Daniel, M.S. Hacid, F. Casati, B. Benatallah. Message Correlation and Web Service Protocol Mining from Inaccurate Logs. [MYD+10];
- K. Musaraj, F. Daniel, M.S. Hacid, F. Casati, DOBS: A Dynamic Model Behavior Simulator for Model Assessment and Mining [MDH+10];
- D. Devaurs, K. Musaraj, F. De Marchi, M.S. Hacid, Timed transition discovery from web service conversation logs [DMD+08].

6.1. Internal documents

- [D5.3] “Final goal-oriented data model”, ver. 1.0 of 2009-07-31.
- [D5.4] “Reasoning mechanisms to support the identification and the analysis of problems associated with user requests”, ver. 1.0 of 2009-12-15.
- [D5.5] “Compliance governance dashboard”, ver. 1.0 of 2010-12-24.
- [D7.1] “Public Web-Site”, <http://www.compas-ict.eu>
- [DoW] “Description of Work”, ver. 15 of 2007-09-25.

6.2. External documents

- [MYD+10] K. Musaraj, T. Yoshida, F. Daniel, M.S. Hacid, F. Casati, B. Benatallah. Message Correlation and Web Service Protocol Mining from Inaccurate Logs. Proceedings of ICWS'10, IEEE, 2010.
- [MDH+10] K. Musaraj, F. Daniel, M.S. Hacid, F. Casati, DOBS: A Dynamic Model Behavior Simulator for Model Assessment and Mining. Sigmod Records, *Submitted*, 2010.
- [DMD+08] D. Devaurs, K. Musaraj, F. De Marchi, M.S. Hacid, Timed transition discovery from web service conversation logs, Proceedings of CAISE'08, ACM, 2008.
- [EY09] E. Young. European fraud survey 2009. Available at: http://www2.eycom.ch/publications/items/fraud_eu_2009/200904_EY_European_Fraud_Survey.pdf
- [MMZ05] E. D. Maria, A. Montanari, M. Zantoni. Checking workflow schemas with time constraints using timed automata. Proceedings of OTM Workshops, 2005.
- [SF10] Source Forge. Available at <http://sourceforge.net/>
- [MM10] Matlab, Mathworks. Available at <http://www.mathworks.com>