

The background of the top section features a large, stylized, light gray 'SPIM' logo. The 'S' is a thick, curved line. The 'P' is a thick, vertical line. The 'I' is a thin, vertical line. The 'M' is a thick, angular shape.

SPIM

Thèse de Doctorat



école doctorale **sciences pour l'ingénieur et microtechniques**
U N I V E R S I T É D E B O U R G O G N E

Construction, Enrichment and Semantic Analysis of Timelines

Application to Digital Forensics



YOAN CHABOT

SPIM

Thèse de Doctorat



école doctorale sciences pour l'ingénieur et microtechniques
U N I V E R S I T É D E B O U R G O G N E

THESIS presented by

YOAN CHABOT

in fulfilment of the requirements for the
Doctor of Philosophy Degree of
University of Burgundy

Field : **Computer Science**

Construction, Enrichment and Semantic Analysis of Timelines Application to Digital Forensics

Research Units :

Electronics Laboratory, Computer Imaging - UMR CNRS 6306,
UCD School of Computer Science and Informatics

Publicly defended on 2015 November in front of a jury composed of :

SÉBASTIEN FERRÉ	Reviewer	Lecturer, IRISA, Rennes
MICHEL HASSENFORDER	Reviewer	Professor, MIPS, Mulhouse
NEIL HURLEY	Reviewer	Lecturer, University College Dublin
CHRISTOPHE NICOLLE	Co-director	Professor, University of Burgundy, Dijon
TAHAR KECHADI	Co-director	Professor, University College Dublin
AURÉLIE BERTAUX	Supervisor	Lecturer, University of Burgundy, Dijon

Acknowledgements

This thesis would not have been possible without the encouragement and the help of several people I want to thank here. I would first like to warmly thank the jury members for agreeing to give attention to my thesis and to evaluate my work.

This thesis could not be brought to completion without financial supports. I want to thank the University College Dublin and the Burgundy region for allowing this partnership between France and Ireland to emerge. The trust that was placed in me and my work honors me. The months spent in Dublin was a particularly rewarding experience both personally and professionally.

Then I want to express my gratitude to my supervisors for their investment during these three years. I am indebted for the degree of freedom that they gave me during all the research process. Thank you to my first co-director, Christophe Nicolle, who allowed me to participate for seven years in the life and work of the CheckSem team. He was an important source of inspiration and knowledge for my research. His enthusiasm, his pedagogy and his analytical skills were invaluable to me all these years. I would then like to thank my second co-director, Tahar Kechadi, for the welcome I have received in Dublin, for his guidance throughout the research process and his valuable help in proofreading of our publications. I am grateful to my supervisor Aurélie Bertaux. Her unwavering support and dynamism were very helpful throughout the project. I measure my chance to have been supervised by such a considerate and friendly person. I wish her a great academic career and a lot of happiness (and courage!) with her family.

Thanks to all the former members and current members of the CheckSem team. I really enjoyed the working atmosphere combining professionalism and friendliness. I wish the team and all its members an excellent continuation and success in their projects. Thank you especially to Christophe Cruz for offering me my first research experience and for giving me guidance during my training.

Thanks to my friends to be what they are and to have allowed these three years to be a period in my life I will not forget. Thank you for the PhD Nights, the (too) many coffee breaks, the tarot and sports... Thank you especially to Barbelette my eternal friend, Kiril, David, Khalid, Fayrouz, Guillaume, Tarcisio, Thomas, Rafael, Anett and many others... I want to thank my brothers and my sister for being there, simply. I am also thankful to my parents, Cyprien, Prunelle, Jocelyne and Serge for their presence and support. The communicative cheerfulness of my nephews and nieces was invigorating during tough times.

Last but not least, thanks to Séverine for being with me on this journey that is the PhD. Her unwavering support gave me the energy to carry out this project. I thank her for the long evenings spent to read my papers. Her opinion on my work has been a great help to improve the quality of it.

Relevant Publications

The following works have emerged in whole from the work contained in this thesis.

International Journals:

- Chabot, Y., Bertaux, A., Nicolle, C., and Kechadi, T. (2015b). An ontology-based approach for the reconstruction and analysis of digital incidents timelines. *Digital Investigation*
- Chabot, Y., Bertaux, A., Nicolle, C., and Kechadi, T. (2014b). A complete formalized knowledge representation model for advanced digital forensics timeline analysis. *Digital Investigation (Fourteenth Annual DFRWS Conference)*, 11(2):S95–S105

International Conferences:

- Chabot, Y., Bertaux, A., Nicolle, C., and Kechadi, T. (2015a). De la scène de crime aux connaissances: représentation d'évènements et peuplement d'ontologie appliqués au domaine de la criminalistique informatique. In *Extraction et Gestion des Connaissances 2015*
- Chabot, Y., Bertaux, A., Nicolle, C., and Kechadi, T. (2014a). Automatic timeline construction and analysis for computer forensics purposes. In *IEEE Joint Intelligence & Security Informatics Conference 2014 (IEEE JISIC2014)*
- Chabot, Y., Bertaux, A., Nicolle, C., and Kechadi, T. (2014d). Reconstruction et analyse sémantique de chronologies cybercriminelles. In *Revue des Nouvelles Technologies de l'Information*, pages 521–524

Book Chapter:

- Chabot, Y., Bertaux, A., Nicolle, C., and Kechadi, T. (2014c). Event reconstruction: A state of the art. *Handbook of Research on Digital Crime, Cyberspace Security, and Information Assurance*

Abstract

Having a clear view of events that occurred over time is a difficult objective to achieve in digital investigations (DI). Event reconstruction, which allows investigators to build and to understand the timeline of an incident, is one of the most important steps of a DI process. The complete understanding of an incident and its circumstances requires on the one hand to associate each piece of information to its meaning, and on the other hand to identify semantic relationships between these fragments. This complex task requires the exploration of a large and heterogeneous amount of information found on the crime scene. Therefore, investigators encounter cognitive overload problems when processing this data, causing them to make mistakes or omit information that could have a high added value for the progress of the investigation. In addition, any result produced by the reconstruction process must meet several legal requirements to be admissible at trial, including the ability to explain how the results were produced.

To help the investigators to deal with these problems, this thesis introduces a semantic-based approach called SADFC. The main objective of this approach is to provide investigators with tools to help them find the meaning of the entities composing the crime scene and understand the relationships linking these entities, while respecting the legal requirements. To achieve this goal, SADFC is composed of two elements. First, SADFC is based on theoretical foundations, ensuring the credibility of the results produced by the tools via a formal and rigorous definition of the processes used. This approach then proposes an architecture centered on an ontology to model and structure the knowledge inherent to an incident and to assist the investigator in the analysis of this knowledge. The relevance and the effectiveness of this architecture are demonstrated through a case study describing a fictitious investigation.

Résumé

Obtenir une vision précise des événements survenus durant un incident est un objectif difficile à atteindre lors d'enquêtes de criminalistique informatique. Le problème de la reconstruction d'événements, ayant pour objectif la construction et la compréhension d'une chronologie décrivant un incident, est l'une des étapes les plus importantes du processus d'investigation. La caractérisation et la compréhension complète d'un incident nécessite d'une part d'associer à chaque fragment d'information sa signification passée, puis d'établir des liens sémantiques entre ces fragments. Ces tâches nécessitent l'exploration de grands volumes de données hétérogènes trouvés dans la scène de crime. Face à ces masses d'informations, les enquêteurs rencontrent des problèmes de surcharge cognitive les amenant à commettre des erreurs ou à omettre des informations pouvant avoir une forte valeur ajoutée pour les progrès de l'enquête. De plus, tout résultat produit au terme de la reconstruction d'événements doit respecter un certain nombre de critères afin de pouvoir être utilisé lors du procès. Les enquêteurs doivent notamment être en capacité d'expliquer les résultats produits.

Afin d'aider les enquêteurs face à ces problèmes, cette thèse introduit l'approche SADFC. L'objectif principal de cette approche est de fournir aux enquêteurs des outils les aidant à restituer la sémantique des entités composant la scène de crime et à comprendre les relations liant ces entités tout en respectant les contraintes juridiques. Pour atteindre cet objectif, SADFC est composé de deux éléments. Tout d'abord, SADFC s'appuie sur des fondations théoriques garantissant la crédibilité des résultats produits par les outils via une définition formelle et rigoureuse des processus utilisés. Cette approche propose ensuite une architecture centrée sur une ontologie pour modéliser les connaissances inhérentes à la scène de crime et assister l'enquêteur dans l'analyse de ces connaissances. La pertinence et l'efficacité de ces outils sont démontrées au travers d'une étude relatant un cas d'investigation fictive.

CONTENTS

1	Introduction	3
1.1	Research Problems	4
1.2	Goals	5
1.3	Contributions	6
1.4	Thesis Structure	7
2	Introduction to Computer Forensics and Event Reconstruction	13
2.1	Legal Process	14
2.1.1	Legal Case and Protagonist	14
2.1.2	Footprint and Evidence	14
2.2	Digital Forensics	16
2.2.1	Resolution of Criminal Cases Using Science	16
2.2.2	Digital Forensics and Cybercrimes	17
2.2.3	Quantifying Cybercrime: A Study of the Situation in France	18
2.3	From Digital Investigation to Event Reconstruction	19
2.4	Challenges Related to Event Reconstruction	21
2.5	Conclusion	23
3	Event Reconstruction	25
3.1	Overview of Event Reconstruction Approaches	26
3.1.1	ECF: Event Correlation for Forensic Purposes	26
3.1.2	Auto-ECF	27
3.1.3	FORE: Forensics of Rich Events	27
3.1.4	Finite State Machine Approach	28
3.1.5	Zeitline	28

3.1.6	A Framework for Post-Event Timeline Reconstruction Using Neural Networks	29
3.1.7	FACE: Forensics Automated Correlation Engine	29
3.1.8	CFTL: Cyber-Forensic TimeLab	29
3.1.9	Plaso	30
3.1.10	PyDFT	30
3.2	Data Volume	31
3.2.1	Automated Timeline Generation	32
3.2.2	Analysis Tools	32
3.2.3	Visualisation Tools	33
3.2.4	Synthesis	34
3.3	Heterogeneity	34
3.3.1	Management of Multiple Sources	35
3.3.2	Knowledge Representation for Event Reconstruction	36
3.3.2.1	Data Formats	37
3.3.2.2	Data Models	38
3.3.3	Synthesis	39
3.4	Legal Requirements	39
3.5	Discussion	42
4	Digital Investigation Process Model	45
4.1	Investigation Process Model: Needs and Challenges	45
4.2	Digital Investigation Process Model: a State of the Art	46
4.2.1	Pre-Investigation	47
4.2.2	Investigation Core	49
4.2.3	Post-Investigation	52
4.2.4	Other Works	53
4.3	Discussion	57
4.4	Conclusion of the State of the Art	58
5	Approach	61
5.1	Semantic Analysis of Digital Forensics Cases	62

5.2	Contribution of Semantics for Analysis of Incidents	63
5.2.1	Knowledge Model for Event Reconstruction	66
5.2.2	Architecture	68
5.3	Credibility and Traceability	69
5.4	Conclusion	70
6	Formalisation of Event Reconstruction	73
6.1	Formal Definition of Crime Scenes	74
6.1.1	Crime scene	74
6.1.2	Time and Location	75
6.1.2.1	Time	75
6.1.2.2	Location	76
6.1.3	Entities Composing a Crime Scene	77
6.1.3.1	Event	78
6.1.3.2	Subject	79
6.1.3.3	Object	79
6.2	Formal Operators for Event Reconstruction	80
6.2.1	Specification of Events	80
6.2.2	Timeline Analysis and Incident Scenario Extraction	82
6.3	Process Model for the Automation of Digital Investigations	85
6.3.1	Specification of PaLADIN	86
6.3.1.1	Pre-Investigation	88
6.3.1.2	Investigation Core	88
6.3.1.3	Post-Investigation	91
6.4	Case Study	92
6.5	Conclusion	99
7	Ontology	103
7.1	Overview	103
7.2	Time and Location	105
7.3	Common Knowledge About the Incident	109

7.4	Specialised Knowledge About the Incident	113
7.5	Reproducibility of the Investigation Process	113
7.6	Conclusion	121
8	Architecture	123
8.1	Overview	123
8.2	Extraction	125
8.2.1	Information Sources	126
8.2.2	Information Extraction of Digital Footprints Using the Plaso Toolbox .	127
8.2.3	Advanced Information Extraction and Serialisation	129
8.3	Instantiation	133
8.4	Analysis	136
8.4.1	Knowledge Enhancement	136
8.4.1.1	Deduction of the Involvement of a User in an Event	137
8.4.1.2	File Classification	141
8.4.2	Timeline Analysis	141
8.5	Data Visualisation	145
8.6	Conclusion	149
9	Experiments	151
9.1	Data Sets	152
9.2	Quantitative Evaluation	153
9.3	Qualitative Evaluation	154
9.4	Conclusion	158
10	Conclusion	163
10.1	Contributions of the Thesis	164
10.2	Issues and Future Work	166
A	Semantic Web	171
A.1	A Brief Introduction to the Semantic Web	171
A.2	Semantic Web Architecture	172

A.3	Knowledge Base and Ontology	173
A.4	Ontology Languages	175
A.4.1	URI and Namespaces	175
A.4.2	RDF/RDFS	175
A.4.3	OWL	177
A.5	Manipulation of Ontologies and Reasoning	178
A.5.1	SPARQL	178
A.5.2	Reasoning	179
B	G-OWL: Graphical Web Ontology Language	181
B.1	G-OWL Symbols	182
B.2	Axioms of Classes and Individuals	183
B.3	Axioms of Properties	183
B.4	Constraints on Properties	184
B.5	Conclusion	185
C	Serialisation of the ORD2I Ontology	187
D	Information Sources	197
D.1	Information Sources	197
D.2	Firefox	198
D.3	Chrome	200
D.4	Internet Explorer	203
D.5	Safari	204
D.6	Win Prefetch	205
D.7	Filestat	206
D.8	Google Drive	208
D.9	Link	209
D.10	Recycle Bin	210
D.11	Skype	211
D.12	Java IDX	212
D.13	OLE Compound File	213

LIST OF FIGURES

2.1	Evolution of cybercrimes in France	18
4.1	Integrated Digital Investigation Process (IDIP)	54
4.2	Physical crime scene investigation in IDIP	54
4.3	Digital crime scene investigation in IDIP	54
4.4	Event-based Digital Forensic Investigation Framework (EDFIF)	55
4.5	Evidence searching in EDFIF	55
4.6	Event reconstruction in EDFIF	56
4.7	End to End Digital Investigation (EEDI)	56
5.1	Interactions between the components of the SADFC approach	63
5.2	Semiotic triangle	64
5.3	Event reconstruction process	65
6.1	A complete view of PaLADIN	87
6.2	Pre-investigation	88
6.3	Investigation core	89
6.4	Physical investigation	91
6.5	Phase 1 of the digital investigation	92
6.6	Post investigation	92
6.7	Relationships between events and subjects	96
6.8	Relationships between events and objects	97
7.1	Concepts and properties of the OWL-Time ontology used in ORD2I	107
7.2	Concepts modelling locations in ORD2I	107
7.3	Properties modelling locations in ORD2I	108
7.4	Classes and object properties of the CKL	109

7.5	Hierarchy of object properties of the CKL	110
7.6	Datatype properties of the CKL	111
7.7	Hierarchy of classes of SKL (first part)	114
7.8	Hierarchy of classes of SKL (second part)	115
7.9	Datatype properties of SKL	116
7.10	Classes and object properties of the TKL	117
7.11	Datatype properties of the TKL	118
8.1	Connexion between PaLADIN and ANNALIST	124
8.2	Overview of ANNALIST	125
8.3	Extraction layer	125
8.4	Mapping from footprints to ORD2I	134
8.5	Patterns used to deduce new relationships between users and events . . .	137
8.6	Correlation graph of events given in Listing 8.1 (solid edges (<i>score</i> \geq 0.6), dashed edges (<i>score</i> \geq 0.4 && <i>score</i> $<$ 0.6))	145
9.1	Execution times of the different steps of the process	154
9.2	Correlation graph viewer: wide zoom on the events composing the case . .	157
9.3	Correlation graph viewer: close view on the chain of events related to con- tagiomalware.exe	159
A.1	Semantic web stack	172
A.2	Representation of facts using RDF/RDFS	176
B.1	Symbols of entities in G-OWL	182
B.2	Symbols of properties in G-OWL	183
B.3	Axioms of properties in G-OWL	184
B.4	Constraints on properties in G-OWL	184

LIST OF TABLES

3.1	Evaluation of event reconstruction approaches regarding the problems related to data volume	34
3.2	Evaluation of event reconstruction approaches regarding the problems related to heterogeneity	39
3.3	Evaluation of event reconstruction approaches regarding the compliance with the legal requirements	42
4.1	The eighteen process models studied in our state of the art	47
4.2	Pre-investigation	47
4.3	Investigation core	50
4.4	Post-investigation	53
6.1	Allen algebra	76
6.2	Correlation scores for events from Listing 6.1 (score > 0.4)	97
7.1	Namespace prefixes used in ORD2I	105
8.1	Identification of the involvement of users using knowledge enhancement . .	138
8.2	Correlation scores for events from Listing 8.1 (threshold > 0.4)	143
8.3	Results of the query retrieving information about all webpages visited by the user named UserX	147
8.4	Results of the query giving the list of all the events extracted using Plaso . .	148
9.1	Description of the data sets: types of events and numbers of events for each type	152
9.2	Quantification of processed data volumes through the reconstruction and analysis process	153
9.3	Execution times of the different stages of the process	153
9.4	Results of the query retrieving all the executables	156

9.5	Results of the query retrieving information about events related to contagiomalware.exe	157
D.1	Information sources supported by Plaso and ANNALIST	198

LISTINGS

6.1	Footprints collected in the crime scene using the Plaso toolbox	93
7.1	Turtle serialisation of temporal information related to the visit of a webpage	105
7.2	Turtle serialisation of location information	108
7.3	Knowledge about the download of a file, stored in the CKL	112
7.4	Information about the objects stored in the SKL	118
7.5	Traceability of the information handled by the TKL	120
8.1	Data produced by <i>log2timeline</i> and formatted using <i>psort</i>	127
8.2	XML serialisation	131
8.3	Turtle serialisation of knowledge generated by the instantiation of the CKL and SKL layers of ORD2I	135
8.4	Turtle serialisation of knowledge generated by the instantiation of the TKL layer of ORD2I	136
8.5	SPARQL query retrieving information about the user involved in each event	138
8.6	Knowledge inserted in ORD2I for a deduction	140
8.7	Knowledge about the correlation between <i>ord2i:ApplicationPrefetch82</i> and <i>ord2i:FileDownload70</i>	144
8.8	SPARQL query retrieving information about all webpages visited by the user named UserX	147
8.9	SPARQL query giving the list of the events extracted using Plaso	148
9.1	SPARQL query used to retrieve all the executables	155
9.2	SPARQL query retrieving information about events interacting with conta- giomalware.exe	156
A.1	Representation of facts using RDF/RDFS and Turtle	176
A.2	Example of selection using SPARQL	178
A.3	Example of insertion of new facts using SPARUL	178
A.4	Example of deletion of facts using SPARUL	179
C.1	Turtle serialisation of the ORD2I Ontology	187

D.1	Information extracted from Firefox using Plaso	199
D.2	Information extracted from Google Chrome History using Plaso	200
D.3	Information extracted from Google Chrome Cookies using Plaso	201
D.4	Information extracted from Google Chrome Cache using Plaso	202
D.5	Information extracted from Internet Explorer using Plaso	203
D.6	Information extracted from Safari using Plaso	205
D.7	Information related to Win Prefetch extracted using Plaso	205
D.8	Information extracted from the file system using Plaso	207
D.9	Information extracted from Google Drive using Plaso	208
D.10	Information related to the links extracted using Plaso	209
D.11	Information extracted from the recycle bin using Plaso	210
D.12	Information extracted from Skype using Plaso	211
D.13	Information extracted from Java Web Start using Plaso	212
D.14	Information related to OLECF documents extracted using Plaso	213

Glossary

Knowledge Base Structure providing, on the one hand, means to represent facts, and, on the other hand, tools to reason on this knowledge. It is composed of two elements: a schema, called TBox, and individuals instantiating this schema, called ABox.

Semantic Web Field providing methods for structuring, linking and sharing knowledge on the Web.

Knowledge Engineering Field developing techniques for the construction, the maintenance and the use of knowledge-based systems.

Integrity Characterises a proof that has not been altered or damaged before its study.

Reproducibility The principle of reproducibility is used to express the ability to reproduce the process used to lead to a given conclusion. The reproducibility allows a court to fully-understand the path used to reach each conclusion which is presented during a trial.

Investigation process model Definition of the nature and of the scheduling of the steps composing an investigation process.

Cybercrime Concept encompassing all criminal offences that may be committed against or with the help of information and communications technologies.

Ontology Formal structure defining the entities, the properties and the logical constraints of a knowledge domain.

Heterogeneity A data set is heterogeneous if the pieces of information it contains have a different syntax, different semantics or a different temporality.

Admissibility An evidence is admissible if it can be used at trial to prove or disprove the facts.

Credibility The notion of credibility refers to the situation where a court questioned the veracity of an evidence. The level of credibility of an evidence is directly related to the level of accuracy and verifiability of the methods used to produce the evidence and the source which produced the evidence.

Forensic Application of scientific methods during investigations to assist the justice in determining the circumstances of incidents.

Digital forensics Field (also called computer forensics) providing investigators with tools and methods to handle cases in which information and communications technologies

have been a mean to commit an infraction or a target of a malicious act.

Footprint Sign left by the presence of an entity or a past action that has occurred where the footprint is discovered (Ribaux, 2013). A footprint is a physical or digital item left by the protagonists in the crime scene during their activities.

Proof Piece of information convincing of the existence or non-existence of a fact. When an evidence is declared as admissible, it can then be considered for judging and becomes a proof. A proof can be used as incriminating evidence to prove the guilt of a part, as exculpatory evidence to prove the innocence of a part or as an element that questions the integrity of other evidence.

Investigation Meticulous research conducted to collect evidence in order to prove the guilt or innocence of a person or an organisation. In this work, we distinguish physical investigations (i.e. an investigation of a physical crime scene) of digital investigations (i.e. an investigation of a digital crime scene).

Timeline Chronology retracing the history of events within a given time frame.

Event reconstruction Step of an investigation enabling the investigators to have a global overview of the events occurring before, during and after a given incident. The event reconstruction can be seen as a process taking as input a set of footprints and outputting a timeline of the events describing the case. In this thesis, the terms "event reconstruction" and "reconstruction of events" are used to refer to the construction of a timeline and its analysis.

Legal case Situation following an incident whose gravity and complexity vary and requiring an investigation.

Criminal case These cases concern incidents that can be reprimanded by a sentence under the law.

Crime scene Place where an incident has occurred. Depending on the nature of the incident, a crime scene can be composed of several physical and digital crime scenes. A physical crime scene is a physical environment containing physical evidences related to an incident. A digital crime scene is defined as a virtual environment created by hardware and software and containing digital evidences related to an incident. A digital crime scene can be a computer, a phone or any electronic devices.

ACRONYMS

- **ADFM:** Abstract Digital Forensic Model.
- **ANNALIST:** Architecture for the recoNstruction and the Analysis of digital IncidentS Timelines.
- **Auto-ECF:** Automatic Event Correlation for Forensic Purposes.
- **CFFTPM:** Computer Forensics Field Triage Process Model.
- **CFIP:** Computer Forensic Investigative Process.
- **CKL:** Common Knowledge Layer.
- **CPMICF:** Common Process Model for Incident and Computer Forensics.
- **CyboX:** Cyber Observable Expression.
- **DFAX:** Digital Forensic Analysis eXpression.
- **DFIPM:** New Digital Forensics Investigation Procedure Model.
- **DFMMIP:** Digital Forensic Model based on Malaysian Investigation Process.
- **DFXML:** Digital Forensics Extensible Markup Language.
- **ECF:** Event Correlation for Forensic Purposes.
- **EDFIF:** Event-based Digital Forensic Investigation Framework.
- **EEDI:** End to End Digital Investigation.
- **EIDIP:** Enhanced Integrated Digital Investigation Process.
- **EMCI:** Extended Model of Cybercrime Investigation.
- **FACE:** Forensics Automated Correlation Engine.
- **FDFI:** Framework for a Digital Forensic Investigation.
- **FORE:** Forensics of Rich Events.
- **FTK:** Forensic Toolkit.
- **GCFIM:** Generic Computer Forensic Investigation Model.
- **G-OWL:** Graphical Web Ontology Language.
- **HOFDIP:** An Objective-based Framework for the Digital Investigation Process.
- **ICTs:** Information and Communications Technologies.
- **IDIP:** Integrated Digital Investigation Process.
- **LEP:** Event Logical Patterns.
- **MHEI:** Process model for the Automation of Digital INvestigations.

- **ORD2I:** Ontology for the Representation of Digital Incidents and Investigations.
- **OWL:** Web Ontology Language.
- **OWL 2 DL:** Web Ontology Language 2 Description Logic.
- **OWL 2 RL:** Web Ontology Language 2 Rule Language.
- **OWL 2:** Web Ontology Language 2.
- **PaLADIN:** Process model for the Automation of Digital INvestigations.
- **PFP:** Proactive Forensic Process.
- **PROV-O:** Provenance Ontology.
- **PyDFT:** Python Digital Forensic Timeline.
- **RDF:** Resource Description Framework.
- **RDFS:** Resource Description Framework Schema.
- **SADFC:** Semantic Analysis of Digital Forensic Cases.
- **SDFIM:** Systematic Digital Forensic Investigation Model.
- **SKL:** Specialised Knowledge Layer.
- **SPARQL:** SPARQL Protocol and Resource Description Framework Query Language.
- **SPARUL:** SPARQL Update.
- **TKL:** Traceability Knowledge Layer.
- **URI:** Uniform Resource Identifier.

INTRODUCTION

INTRODUCTION

The evolution of new technologies and their uses has led to a sharp and rapid increase of the amount of data produced daily (Gantz and Reinsel, 2012). This increase can be explained by the intensive and widespread use of new technologies in everyday life and by the increase of the number of connected objects and of their storage capacity. This digital revolution has deeply impacted and transformed the work of investigators in solving criminal cases and other incidents. The field of digital forensics aims to respond to these new challenges by providing investigators with tools and methods to handle cases in which the Information and Communications Technologies (ICTs) have been a mean to commit an infraction or a target of a malicious act (e.g. denial of service attack, theft of personal information using phishing¹, credit card fraud, a murderer approaching his victims via social networks, etc.). Digital objects are now at the heart of many cases and the study of footprints generated by these objects has become a crucial and indispensable step in the investigations. Indeed, the study of digital footprints left by the activities of the user (such as web browsing, the interactions with files and their transfers, digital communications, etc.) is an important source of information for the investigators.

During an investigation, the main goal is to reconstruct a timeline (i.e. a chronology retracing the history of events within a given time frame) composed of the events that occurred during the incident, using the information left in the digital crime scenes by the protagonists of the incident (i.e. suspects and victims). Among the steps composing an investigation, the event reconstruction is a critical step, allowing investigators to have a view of past events. In this thesis, the terms "event reconstruction" and "reconstruction of events" are used to refer to the construction of a timeline and its analysis. This process aims to determine the nature of the incident, its characteristics and the protagonists involved from the description of past events. Although it may result in a timeline inaccurate and incomplete, the construction and the analysis of timelines help answer questions about past actions. To construct a timeline, the investigators start by collecting all the digital objects that can be found in the crime scene and then extract and copy the data contained in them. The disk images are then scanned and analysed in search of foot-

¹Technique used by fraudsters to extort personal information by pretending to represent trusted institutions

prints left by the activities of the protagonists. The extraction of these footprints enables the identification and the characterisation of the events that occurred in the past. Following the construction of the timeline regrouping all these events, the interpretation and the analysis of it enable the investigators to view the interactions between the protagonists, to understand the circumstances of the incident and finally to determine the responsibilities of each protagonist. The investigators can then draw conclusions supported by evidences to prove the guilt or the innocence of the suspects.

1.1/ RESEARCH PROBLEMS

The massive use of ICTs significantly increases the amount of data generated by each person and it also causes the presence of a growing number of digital objects on crime scenes. Therefore, the investigators are confronted with the need to analyse large amounts of data to complete the reconstruction of events. Moreover, these data are scattered across many heterogeneous sources (different devices, data produced by different software or services, etc.) making the reconstruction very tedious. These problems inherent in the reconstruction of events are very similar to those encountered in the field of data mining. This field offers methods and algorithms to extract knowledge from massive amounts of heterogeneous data. It enables the uncovering of hidden information from the gangue of data and introduces tools to describe the distribution of values of a variable, to identify groups of similar entities, or identify associations between the values of the variables. These knowledge extraction methods, called descriptive analytics, are intended to summarise, synthesise and classify the information.

Nevertheless, beyond a simple browsing of data and an extraction of a small relevant collection of pieces of information, the problem of reconstruction of events requires to clearly identify the inherent meaning of each piece of data, collected in each footprint of the crime scene. A footprint is a sign left by the presence of an entity or a past action that has occurred where the footprint is discovered (Ribaux, 2013). Each footprint belongs to a past that the investigators are trying to reconstruct to understand what happen during the incident. Therefore, each footprint must be studied to restore its original meaning (e.g. a footprint collected in a user's browsing history may, for example, correspond to the download of a file from a remote server). In addition, like in a jigsaw puzzle, the event reconstruction also requires to place each of these pieces of information into a bigger picture, forming the scenario of the incident. Thus, the reconstruction of the scenario of the incident involves the identification of the semantic relationships connecting the various fragments collected in the crime scene, in order to understand the role played by each of them in the incident. Moreover, like any digital forensics methods, the reconstruction of events is governed by strict rules ensuring the admissibility of the results at trial. Because the results of the process are used as evidence, they must achieve a high degree

of accuracy and truthfulness and also must be fully explained by the investigators. In particular, each conclusion must be supported by evidence rigorously collected, giving it full credibility and the reasoning used to draw a conclusion must be documented.

The ability of data mining techniques to address these issues is hampered by the fact that these are mainly statistical methods that do not take into account the semantics of data. In addition, the use of such techniques do not always allow to correctly explain the results outputted. This is particularly true for the techniques based on supervised or unsupervised learning such as neural networks (Khan et al., 2007) that behave like black boxes where many variables remain unknown and unexplained to the user.

To address the difficulties faced by the investigators during the event reconstruction process, this thesis provides answers to the following two research problems:

- This research work first focuses on the **characterisation of incidents** (i.e. determining the causes, the effects and the circumstances of them) by a **semantic reconstruction and a semantic analysis of the activities (and the relationships between them) composing the incidents**. The discovery of knowledge from large and heterogeneous volumes of data is a research problem also addressed in the area of big data (Value) (James et al., 2011).
- Second, this thesis focuses on the development of **mechanisms to ensure that the scenarios of incidents reconstructed by the tools can be used as evidence in trial**. This research problem can be likened to the search for Veracity in the area of big Data (Zikopoulos et al., 2011; Hitzler and Janowicz, 2013; Emani et al., 2015). This emerging research theme seeks to prove the validity of the conclusions produced during the extraction of knowledge.

1.2/ GOALS

To contribute to the improvement of event reconstruction techniques, we have set in this thesis seven objectives that appear as important steps towards the resolution of the two research problems above:

- Because of the volume of data that can be found in a crime scene, building a timeline representing the events that took place in the crime scene is a complex and tedious task for investigators. To get a complete view of the events that occurred in the past, a lot of information has to be managed and integrated in a single timeline. Therefore, it is necessary to help the investigators in handling large data volumes by 1) providing tools to **generate automatically the timeline** from all the information that can be found in a crime scene. Moreover, once the integration of information is complete, the analysis of the timeline is needed to get a better understanding of the incident and to draw conclusions. These tasks are made complex because of the

difficulty in handling the knowledge, the difficulty to identify logical connections between events and the non-intuitiveness of the consultation of the information. This leads to a cognitive overload that prevents the investigators to do their work in the time allotted by justice and may even cause them to make mistakes or miss out on essential information. Thus, a priority is 2) to **assist the investigators in the analysis of the timeline** and 3) to **facilitate the understanding and the manipulation of the knowledge** collected during the investigation by making available ergonomic and intuitive visualisation tools.

- Due to the dispersion of the data in the crime scene and because of the syntactic and semantic heterogeneity, it is complicated to get a comprehensive and precise view of the incident uniting all information that can be collected from the crime scene. Such a view is needed to enable investigators to understand the past situation and draw correct conclusions. To ensure a complete and accurate reconstruction of events, it is required 4) to propose a storage model which **federates all the knowledge collected from the crime scene** in a single view and 5) to **solve the technical problems related to the heterogeneity of data** by introducing **extraction mechanisms able to collect the information** from the crime scene to instantiate the previous model.
- To be admissible in a court of law, the results produced at the end of the reconstruction of events must meet certain legal requirements including 6) the **credibility of the results** produced by the tools and 7) the **reproducibility of the methods** of investigation (Baryamureeba and Tushabe, 2004). It is particularly necessary to ensure that all evidences presented at trial are credible and that the methods used to produce evidence are reproducible and did not alter the objects found in the crime scene.

1.3/ CONTRIBUTIONS

The main contribution of this thesis is a semantic-based approach named Semantic Analysis of Digital Forensic Cases (SADFC). This latter is a synergy of several elements designed to give answers solving the problems mentioned above. To solve the issues inherent to the quantity of data and its heterogeneity, a promising perspective consists in using a precise and reliable representation, enabling to structure data on the one hand, and to standardise their representation on the other hand. The use of a structured and formal knowledge representation makes the work of automated processes easier by making information understandable by machines and gives to investigators an easy way to query, analyse and visualise the information. Therefore, the first contribution of this work is an **ontology** providing the means to store the knowledge extracted from a crime scene. This ontology, thanks to the expressiveness of the Web Ontology Language 2 (OWL 2), constitutes an accurate and complete picture of the events that occurred before, during and after

a digital incident. It is divided into three layers storing generic knowledge about events, specialised and technical characteristics of objects and information on tasks performed by the investigators during the investigation. In order to make available to investigators the capabilities needed to carry out the reconstruction of events, this ontology is integrated into a **software architecture** which is the second contribution of this thesis. This architecture consists of four modules to complete the information extraction from disk images found in a crime scene, the instantiation of the ontology with this knowledge and the analysis and the visualisation of this knowledge.

To meet the legal requirements, the SADFC approach is based on theoretical foundations. The third contribution of this thesis is **formal operators** for the reconstruction of events and their analysis. These operators are based on formal definitions of a crime scene and the entities composing it. To ensure the reproducibility of the investigation process, the fourth contribution of this thesis is an **investigation process model** defining precisely the steps composing our event reconstruction process.

1.4/ THESIS STRUCTURE

This manuscript is organised into five parts in addition to this introduction. The first part of the manuscript is a complete state of the art on the issue of the reconstruction of events and related problems. First, Chapter 2 introduces the key concepts of digital forensics, the field of application of this thesis. After introducing the legal process shared by every investigation, the notion of cybercrime is explained and its impact on our society is quantified. The field of computer forensics aims to provide investigators with methods and tools to solve cybercriminal cases. A generic investigation process is especially presented to explain the path used by investigators to carry out a digital investigation. Then, a zoom is made on the problem of event reconstruction, which is an important step of a digital investigation, and also the focus of this research work. The issues related to this process are finally detailed at the end of this chapter. Second, Chapter 3 reviews the existing event reconstruction approaches and evaluates them in the light of the solutions they provide to address the previous issues. Third, Chapter 4 introduces the notion of investigation process model. A digital investigation is a rigorous process governed by strong legal requirements to ensure the credibility and integrity of the evidence. The definition of a process model helps to provide answers to problems related to these legal requirements. Many investigative process models are proposed in the literature. However, there is currently no consensus. This chapter studies the strengths and weaknesses of existing process models in order to subsequently build an adequate model to meet our needs and which can interface satisfactorily with our approach. Indeed, the event reconstruction is one of the steps composing the digital investigation. Interfacing our reconstruction tools in this larger process is an essential step in their development.

After the presentation of the scope of this thesis and the highlighting of the limitations of existing approaches, the second part of the manuscript presents an innovative approach to meet the remaining needs. First, Chapter 5 introduces the SADFC approach to meet these limits using an approach whose central idea is to take advantage of the semantics of data to deal with cognitive overload problems (during the phase of event analysis in particular) while ensuring quality results from a legal point of view. This chapter presents the ideas developed in this approach and briefly introduces the elements composing it. These components are presented in detail in the following chapters. Second, Chapter 6 introduces the theoretical foundations of the SADFC approach. To ensure credibility of results while disambiguating concepts handled in our approach, a formal definition of the entities comprising a crime scene is given. Taking advantage of these definitions, the operation of our event reconstruction approach is then clarified and formal analysis operators are presented. Finally, a new investigation process model is introduced to meet the need for reproducibility and to integrate our event reconstruction approach in the broader context of the digital investigation.

After giving an overview of the approach and its theoretical foundations, the third part of this manuscript shows the implementation of it. First, Chapter 7 presents the central component of the approach: the Ontology for the Representation of Digital Incidents and Investigations (ORD2I). After having motivated the use of an ontology for our case of application, this chapter presents the entities and the relationships composing this ontology. Chapter 8 shows the functioning of the modules of the architecture and their interactions with the ontology. Finally, Chapter 9 demonstrates the relevance of the approach and its ability to meet the needs of investigators. To do so, this chapter first provides a quantitative study assessing the performance of the approach. Second, a case study are proposed to demonstrate the usefulness of the approach on a fictional scenario with characteristics and issues close to those encountered in real world scenarios.

After a conclusion reminding the strengths of the proposed approach, its limitations and the future works planned to improve it, appendices are proposed to complement the vision of the reader on the work presented in this thesis. Appendix A introduces the field of semantic web, and more generally the field of knowledge engineering. The techniques of the semantic web stack used in this research work are presented including Uniform Resource Identifier (URI), the Resource Description Framework (RDF), the Resource Description Framework Schema (RDFS), the Web Ontology Language (OWL), reasoning and query languages. Appendix B presents the Graphical Web Ontology Language (G-OWL), a graphical language for representing OWL ontologies. This formalism is used in this thesis to present the ontology. The choice of this formalism is motivated in this appendix and its concepts are explained. Appendix C provides a serialisation of ORD2I, the ontology presented in this thesis. This serialisation gives a complete view on the ontology which complements the explanations given in Chapter 7. Indeed, despite the clarity of the G-OWL language used throughout this thesis, it can not substitute for serialisation

in due form of the ontology, especially in terms of accuracy. Finally, to address the heterogeneity of the information on a crime scene, SADFC is able to process a wide range of information sources. Appendix D presents each of the information sources processed by our approach in particular explaining the potential value of each of them to achieve the objectives of an investigation.

STATE OF THE ART

INTRODUCTION TO COMPUTER FORENSICS AND EVENT RECONSTRUCTION

The research work carried out during this thesis is at the crossroads of the areas of digital forensics and knowledge engineering. The objective of this work is to answer problems inherent to the field of digital forensics using methods and technologies from the field of knowledge engineering and especially the field of semantic web. Digital forensics is a vast and complex area governed by strong constraints related to its legal aspects. This area provides methods to carry out investigations for cases in which ICTs were a target of a crime (i.e. denial of service attack on a server, theft of payment cards or password on a server, etc.) or a tool to commit a crime (i.e. a pedophile can take advantage of instant messaging software and social networks to identify and approach his victim).

The purpose of this chapter is threefold. The first objective is to place the work done during this thesis in its context: the resolution of cases involving ICTs. The second objective is to introduce the reader to the field of digital forensics to enable him to have the knowledge required to read this manuscript. The third and last goal of the chapter is to introduce the problem of event reconstruction. This problem is the main focus of this thesis, which introduces an innovative approach to provide solutions to it.

This chapter is structured in the following way. Section 2.1 introduces the key concepts of the legal process governing every investigation. This section describes in particular the path conducted since the collection of footprints in the crime scene to the final decision at trial. Section 2.2 explains the concept of forensics and computer forensics. This section also details the types of crimes for which investigators rely on digital forensics and characterises the evolution of these crimes through time. Section 2.3 presents the process used to carry out a digital investigation and introduces the problem of event reconstruction. This latter is an important step of every digital investigation and is also the central problem of this research work. Finally, Section 2.4 highlights the scientific and technical

challenges related to the problem of event reconstruction. It should be noted that the legal dimension of this thesis is based on the French legal system.

2.1/ LEGAL PROCESS

Computer forensics is a tool serving justice to solve cases involving digital objects. Before discussing this area, this section introduces generic concepts (not specific to computer forensics) concerning the legal processes governing each investigation, whether led by justice or not (by a private investigation team for example).

2.1.1/ LEGAL CASE AND PROTAGONIST

A legal case¹ is defined as a situation following an incident whose gravity and complexity vary and requiring an investigation. We distinguish two types of cases. First, the criminal cases² concern incidents that can be reprimanded by a sentence under the law. This type of case requires the assistance of justice. Second, the civil and other cases³ focus on incidents and disputes involving private parties. Depending on the type of the case, an investigation is conducted by legal authorities and law enforcement or a private organisation (detectives for example). In the work proposed in this thesis, we ignore the differences between the different types of investigation. The solutions proposed in our work can be used in any investigation compatible with the concepts described in this chapter. In the rest of the manuscript, the term "court" refers to the entity making a judgment in a given case and the term "investigator" refers to police investigators or private investigators involved in the investigation.

Two types of protagonists are involved in a case. First, the complainant⁴ claims that illicit events occurred in the past and that, under the law, the other party must be condemned. On his side, the defendant⁵ challenged or not one or more aspects of the complainant's story to avoid or reduce the sentence of the court. To solve the case, the court has to verify the truthfulness of the facts evoked by each of the parties. For this, the court considers the evidence collected during the investigation.

2.1.2/ FOOTPRINT AND EVIDENCE

As said previously, the evidences provided to the court enable the judges to decide a case by helping them to determine the causes and circumstances of the incident. Before

¹http://en.wikipedia.org/wiki/Legal_case

²<http://www.vie-publique.fr/decouverte-institutions/justice/fonctionnement/justice-penale/qu-est-ce-qu-affaire-penale.html>

³<http://www.vie-publique.fr/decouverte-institutions/justice/fonctionnement/justice-civile/qu-est-ce-qu-affaire-civile.html>

⁴<http://en.wikipedia.org/wiki/Plaintiff>

⁵<http://en.wikipedia.org/wiki/Defendant>

being submitted to the court, an evidence follows a long path requiring a significant rigour. During it, the evidence passes through several states before being presented at trial. The primary state of an evidence is named footprint. A footprint is a sign left by the presence of an entity or a past action that has occurred where the footprint is discovered (Ribaux, 2013). A footprint is a physical or digital item left by the protagonists in the crime scene during their activities. Each footprint belongs to a past that the investigators are trying to reconstruct to understand what happen during the incident. To assist investigators in this task, every footprint contains one or more information about its source (the entity or the action that created it). However, a footprint may be incomplete or imperfect. Thus, this situation can lead to approximate or uncertain reasoning during the investigation. It is therefore important that the reasoning made by investigators from footprints collected can be reviewed and corrected by them, especially when new footprints bring a different perspective and enable new deductions about the incident.

The investigators have two goals during an investigation. The first goal is to identify the footprints that are potentially relevant for the progress of the investigation. This involves being able to distinguish footprints produced by a common activity (not illicit) and footprints produced during activities related to the claimed offence. The second goal of the investigators is to make a path starting from footprints and arriving to the most likely explanation of the circumstances of their presence in the crime scene. This process is called event reconstruction.

When a footprint is collected by investigators to be used in the resolution of a case, it becomes an evidence. It should be noted that an evidence may be a footprint, a composition of several footprints or even the result of a reasoning (deduction) made using one or several footprints. The evidence is then presented to the court under several conditions. An evidence is declared admissible if deemed relevant to the progression of the trial and if it meets several legal rules inherent to the type of evidence. When an evidence is declared as admissible, it can then be considered for judging and becomes a proof. A proof is an information convincing of the existence or non-existence of a fact. A proof can be used as incriminating evidence to prove the guilt of a part, as exculpatory evidence to prove the innocence of a part or as an element that questions the integrity of other evidence. A proof is characterised by two attributes: its relevance and its weight (Gladyshev, 2004). The relevance of a proof is its ability to change (increase or decrease) the probability of a fact. The weight of a proof quantifies the importance of this change. To preserve the relevance and weight of a proof, the integrity of the latter must be preserved throughout the investigation. For this, the collection and the study of the evidence must be conducted in order to not change its state. To attest of this non-change and allow the evidence to be admissible, a custody chain is used to record operations performed by investigators on the evidence. Each custody chain is created as soon as the evidence is collected and contains information such as an identifier of the evidence, the context of the collection (date, place, investigator in charge of the collection) and the history of manipulations

performed on the evidence.

2.2/ DIGITAL FORENSICS

After introducing forensics and digital forensics, this section defines the concepts of cybercrime and quantifies the importance of this phenomenon. This section is based on a report written by the french inter-ministerial working group on the fight against cybercrime and delivered to the highest French authorities in 2014 (Robert, 2014).

2.2.1/ RESOLUTION OF CRIMINAL CASES USING SCIENCE

Forensic is the application of scientific methods during investigations to assist the justice in determining the circumstances of incidents. The areas involved in responding to investigations are numerous and include:

- Basic sciences: physics, biology, chemistry, mathematics, etc.
- Natural sciences: anthropology, geology, mineralogy, etc.
- Human sciences: psychology, sociology, etc.
- Applied sciences: computer science, electronics, medicine, etc.

Among the most commonly used techniques are DNA identification, the analysis of blood, ballistics techniques and autopsies. The use of scientific processes in an investigation allows the evidences produced to benefit from new properties desired by Justice regarding the admissibility of the evidences. The evidence resulting from the use of science has an objective nature as it is produced using techniques based on logic and proven scientific theories rather than human intuitions (Gladyshev, 2004). It should be noted, however, that to be admissible, an evidence produced using science has to be based on scientific processes accepted by the scientific community. Thus, to check the credibility and acceptability of a scientific expertise, standards such as the Daubert standard (Farrell, 1993) in US have been created. These standards aim to control the reliability of the results produced by a given scientific theory. For this, a theory or a technique must meet several criteria to enable the production of admissible evidence in a court:

- It must be tested or should have been tested in the past.
- It must have been submitted to the appreciation of other members of the scientific community, and therefore have been the subject of publications in conference and peer-reviewed journals.
- It must be recognised throughout the scientific community of the area from which it originated.
- The rate of potential errors should be known.

2.2.2/ DIGITAL FORENSICS AND CYBERCRIMES

The democratisation of new technologies and significant changes in the number and variety of equipment and technological tools have made possible a revolution of our society. However, these changes have been accompanied by a significant growth of a type of crime that was still marginal a few years ago: cybercrime. This is a concept encompassing all criminal offences that may be committed against or with the help of ICTs. This thesis focuses on the field of digital forensics which aims to provide methods to assist the investigators in the resolution of cybercrime cases. We distinguish two types of cybercrimes:

- The cybercrimes in which ICTs are the main means and the main target of the offence. This includes the attacks against automated data processing systems (intrusion in the system, alteration or destruction of data), infringements of individual liberties through automated processing and the "preventive" offences (conception, development and dissemination of software tools for use in the context of illegal activities). As examples for this category of cybercrimes, we can mention cyberterrorism, denial of service attacks on network infrastructures or attacks against the e-reputation of an individual.
- The cybercrimes in which ICTs are the main means of an offence to harm a non digital object (person or property). This includes the use of ICTs for sharing illegal contents (child pornography, incitement to terrorism, ethnic or racial hatred, etc.) or to facilitate an offence. This category covers corruption of minors through instant messaging software or social networks, the organisation of activities on jihadist forums, artificial market speculation using fraudulent disinformation techniques, phishing and farming, economic and industrial espionage, intellectual property infringement, money laundering or tax fraud.

The authorities find that the number of cybercrimes and cybercriminals is increasing and that the cyber criminal activities are diversifying (Robert, 2014). Today, there are several types of cybercriminals, depending on their activities:

- The sex offenders involved mostly in acts of child pornography.
- The abusers threatening, insulting, defaming or harassing people via the Internet.
- The cybercrooks seeking personal enrichment through a wide range of scams such as phishing, blocking of a system with a ransom demand, blackmail, etc.
- The cyber mercenaries offering their technical skills and their criminal expertise to individuals or organisations.
- The cyber spies performing intrusions to steal economic, scientific or strategic information.
- The cyber terrorists using the Internet to communicate, organise their activities and recruit new terrorists.

2.2.3/ QUANTIFYING CYBERCRIME: A STUDY OF THE SITUATION IN FRANCE

Cybercrime has become increasingly commonplace in today's world. Crimes committed with the aid of or against digital systems are being reported almost daily and require corporations and governments to spend millions in security systems (Anderson et al., 2012). However, this new phenomenon is difficult to quantify precisely. In the case of France, the authorities argue that statistics compiled by the police and gendarmerie, even made exhaustive, can not accurately describe the magnitude of the phenomenon. Indeed, many Internet crimes do not result in a complaint or even a denunciation. In France, the authorities find a certain reluctance to make complaints, especially when the offence concerns an organisation as they want to preserve their image by limiting communication on cyber incidents affecting them. Regarding individuals, the user may not know that he is a victim of cybercrime or considers that the injury does not justify filing a complaint.

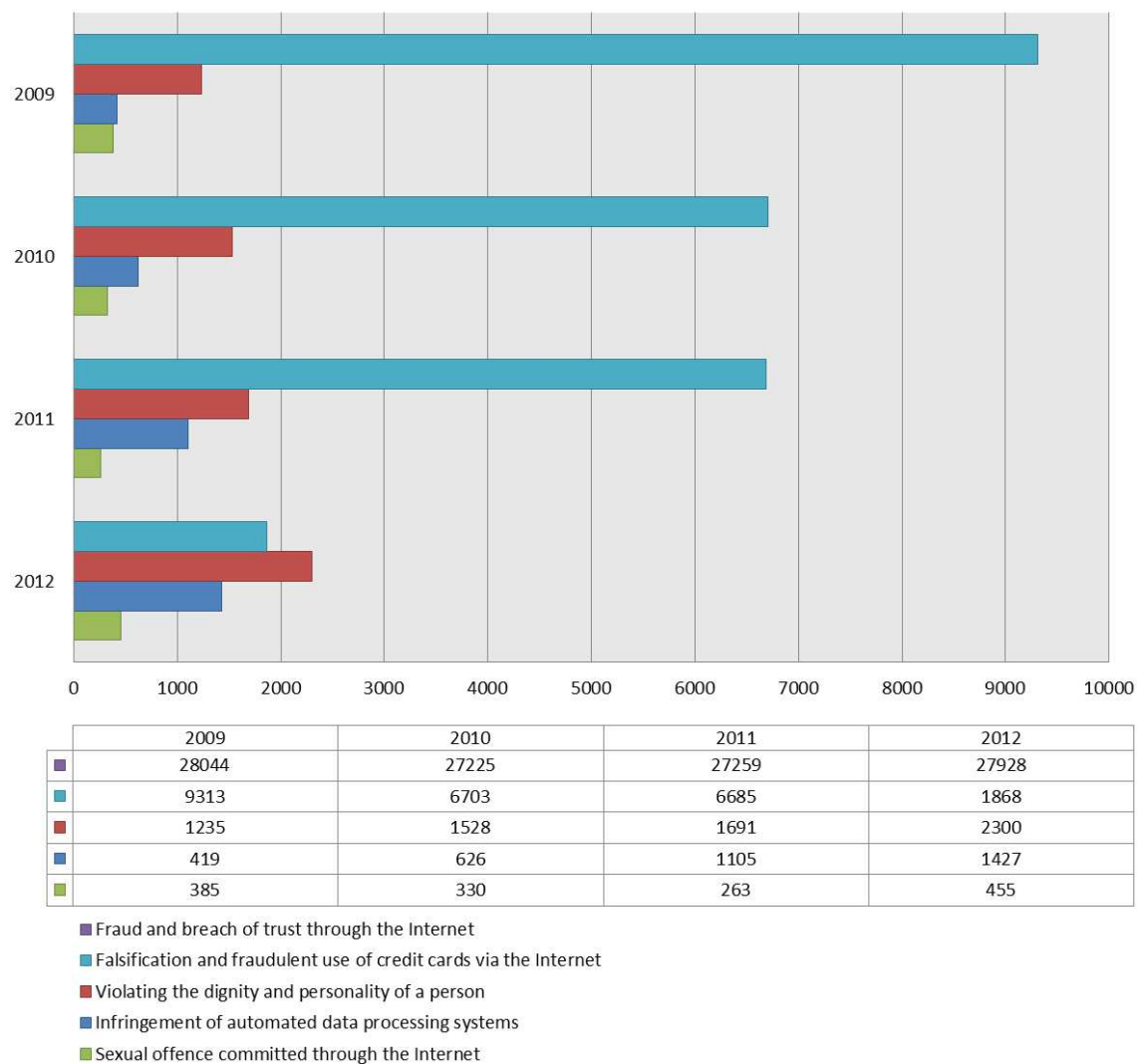


Figure 2.1: Evolution of cybercrimes in France

Despite this, the French authorities have several sources of statistical information, in-

cluding the national police and the gendarmerie. In a study carried out in 2014 (Robert, 2014), trend indicators are given based on the information generated by operational information systems STIC (national police) and JUDEX (national gendarmerie). Figure 2.1 reflects the evolution of several types of cybercrime over the period 2009-2012 (data for subsequent years are not available). It should be noted that this study considers only a part of cybercrimes. In 2012, 84774 cybercrimes were recorded by the police and the gendarmerie. The report notes that the number of cyber criminals is increasing due to the fact that the technologies required to commit cybercrimes have democratised and are now accessible to a large population. In the figure, we note that most types of cybercrime are increasing (infringement of automated data processing systems, violating the dignity and personality of a person, sexual offence committed through the Internet). There is also a stagnation of fraud and breach of trust through the Internet (as the values of this curve are very high, it is not shown in the figure to improve the readability). However, the falsification and fraudulent use of credit cards is decreasing, particularly because of penal policies established to counter this type of offence.

Due to all these issues, digital forensics has become a crucial issue for countries, justice and law enforcement and companies. The increase and diversification of cybercrime require these organisations to invest sums of money ever more important to ensure the safety of individuals and properties. Therefore, the field of digital forensics has grown significantly in recent years and has become an important research area with a large community of researchers, IT professionals and investigators.

2.3/ FROM DIGITAL INVESTIGATION TO EVENT RECONSTRUCTION

These new types of offences require a new type of investigation called **digital investigation**. The rigour of the field requires investigators to follow a digital investigation process consisting of several steps. The use of such a clearly defined process helps investigators to meet the objectives of the investigation, while ensuring the quality of results. Although there is no consensus on the definition of this process, each digital investigation follows a high-level framework, defined in (Palmer, 2001), starting with the identification of an incident and ending with the final decision of the court of justice. This process includes steps enabling to preserve the integrity of evidence, to seize sources of footprints from the crime scene, to examine these sources to find relevant information and finally to analyse this information to be able to make assumptions about the incident:

- Identification: this step marks the beginning of the investigation. The incident is detected through witnesses, surveillance systems or intrusion detection, etc. Investigators are then alerted and briefed on preliminary information possessed about the incident. After preparing their equipment, investigators go to the scene of the incident.

- **Preservation:** the first stage of the investigation, once the investigators are arrived, is to preserve the scene of any possible alterations. This step is crucial to preserve the footprints. The investigators must find ways to stop all processes that modify the state of data without degrading any volatile footprints. This step involves operations such as stopping machines, the live collection of the data contained in RAMs or the isolation of a machine from its network. The preservation step also includes a documentation phase keeping information about the crime scene before the start of the investigative work. The documentation implies taking photos of the scene and the elements composing it and making disk images of machines.
- **Collection:** when the state of the crime scene is frozen, investigators then collect the items that are potentially useful to reach the objectives of the investigation. During a digital investigation, specialised and approved tools and software are used to collect the data, even if the data were hidden or deleted by one of the protagonists of the incident.
- **Examination:** after the collection, the seized items are examined one by one to understand their meaning, and whether each element is really relevant regarding the objective of the investigation. The examination step is a first filtering of the collected footprints.
- **Analysis:** using the remaining data, the analysis step allows the investigators to determine the circumstances of the incident. The main objective is to reconstruct the events composing the incident using the footprints collected from the crime scene (which are signs of past events). After the analysis, investigators reconstruct the chronology of the incident and build a theory detailing the responsibilities of each protagonist and the circumstances of the incident. This theory is supported by evidence derived from footprints.
- **Presentation:** after the formatting of the theory in an admissible format for the court, the results produced by the investigators are presented at the trial. A discussion then begins between the parties to determine the responsibilities of each protagonist and the penalties that must be applied.
- **Decision:** after the deliberations, the court renders its judgement.

Several tools are available to help investigators during the first steps of this process. For example, EnCase or Forensic Toolkit (FTK) can help investigative agents during the collection and the examination of digital objects while preserving their integrity. However, these tools are limited regarding the analysis step, which allows to fully understand what happened during the incident. Collecting evidence and studying its properties is an important part of the investigative process. However, to extract acceptable evidence, it is also necessary to infer new knowledge, such as the causes of the current state of the evidence. (Carrier and Spafford, 2004b). For example, a file illegally modified may be identified during the first steps of an investigation. Although the identification of such an object is interesting, only the analysis phase can help investigators to understand the

causes of this modification. Among all the techniques used during the analysis phase, event reconstruction allows investigators to have a global overview of the events occurring before, during and after a given incident. The story produced as output of this process can answer many questions such as "What happened?" and "Why these events took place?". The field of event reconstruction aims at solving this issue. Event reconstruction is "the process of identifying the underlying conditions and reconstructing the sequence of events that led to a security incident" (Jeyaraman and Atallah, 2006). Due to the importance of this phase and the lack of tools to carry it, we chose to focus on the event reconstruction in this thesis.

2.4/ CHALLENGES RELATED TO EVENT RECONSTRUCTION

Event reconstruction has many issues, which are directly related to the size of the data, digital investigation process complexity, and IT infrastructures challenges. While some of these challenges have been a focus for many researchers and developers for the last decade, the **size of data volumes** (Richard III and Roussev, 2006), the **heterogeneity** of data and the need to meet **legal requirements** are still very challenging. These three issues are discussed in detail in this section.

Today, investigators are facing ever greater data volumes on digital crime scenes during investigations. The growth of the volume of data is caused by several factors. First of all, digital devices are more and more present in our daily lives which increases the number of devices owned by each person and therefore the number of devices found in the crime scenes. In addition, the frequency of use, the new uses and the increase of storage capacity of the digital devices have caused the increase in the quantity of data stored by each device. The very large amount of data which investigators face introduces many challenges at every phase of the digital forensic process; from the data collection to the interpretation of the results. It makes the analysis very complex and tedious, even causing cognitive overload. Information potentially relevant to reach the objectives of the investigation is lost in the amount of data, making the investigation difficult. For example, the Plaso toolbox (Gudhjonsson, 2010), which produces timelines from hard disk image seized from a crime scene, can identify thousands of events from a wide range of sources (Apache logs, Skype conversations, Google Chrome history, Windows event logs, etc.) from an image of only a few gigabytes. In conclusion, to deal with this issue, an event reconstruction approach should propose solutions to allow efficient processing of information and solutions for data retrieval and visualisation in a clear and intuitive way for investigators. Moreover, the automated extraction from a large number of sources lead to the creation of large timeline that are difficult to read, interpret and analyse for the investigators. To assist the investigators during this phase, event reconstruction approaches have to provide analysis tools carrying out all or part of the reasoning and visualisation

tools presenting data in a clear and intuitive way.

Cybercrimes require investigators to handle a new type of footprints, called digital footprints. A digital footprint is information stored or transmitted in a digital form. A footprint can have many forms: a digital image, an email, a log file, a digital document, a data fragment from the RAM of a system or from a disc, etc. The use of digital footprints in a case is often complex as they are spread across the crime scene in multiple sources of footprints (software logs, recycle bin, web histories, etc.). For these reasons, the second important issue of event reconstruction is the heterogeneity of footprints. During an investigation, many sources of information potentially relevant can be used, including web browser histories, windows event log, file system, logs of various software, etc. In order to collect all necessary information to get a true and accurate picture of the incident, the scenario reconstruction approaches must be able to extract the information in all of these sources and process them appropriately. This leads to the development of an automated information processing approach which is able to extract knowledge from these heterogeneous sources. In addition, once extracted, this knowledge should be federated within the same model to facilitate their interpretation and future analysis.

The last challenge of event reconstruction is to ensure that the results produced by the tools will be admissible in a court of law. As (Baryamureeba and Tushabe, 2004) emphasises, all approaches proposed to carry out event reconstruction have to satisfy some key requirements such as credibility and reproducibility of the digital evidence. First, the notion of credibility refers to the situation where a court questioned the veracity of an evidence. The level of credibility of an evidence is directly related to the level of accuracy and verifiability of the methods used to produce the evidence and the source which produced the evidence. In (Stephenson, 2003), the authors explain that "if we wish digital forensics to be considered as scientifically valid, we must show that our tools, methods and techniques are defensible, both from a technical and scientific perspective and from the perspective of the law." To reach this objective, in recent years, the protagonists of digital forensics tried to move from investigative techniques that are based on the investigators' experience, to techniques based on proven theories. Second, the principle of reproducibility is used to express the ability to reproduce the process used to lead to a given conclusion. The reproducibility allows a court to fully-understand the path (made of data handling steps, reasoning steps, etc.) used to reach each conclusion which is presented during a trial. It is therefore necessary to be able to provide clear explanations about the way each evidence is produced. In addition, one has to ensure that the tools used do not modify the data collected in the crime scenes. Thus, it is necessary to develop tools that extract evidence, while preserving the integrity of the data. To address these three issues, we believe that the following criteria are crucial for event reconstruction approaches: the use of a theoretical model to support the proposed approach, the ability to maintain the data integrity, the provision of mechanisms able to generate explanations about reasoning and a formal and standard definition of the reconstruction

process (defining the steps of the investigative process).

It should be noted that each jurisdiction of each country has its own laws and its own legal system. This thesis offers generic solutions that are not rooted in any of these jurisdictions. The problems of credibility and reproducibility of the results that are addressed in this work are universal issues, relevant for all jurisdictions. The ability to support the results of the investigation by logic and clear explanations is interesting for any investigative team, regardless of the country in which this team works. The main idea of the work presented in this manuscript is to assist the investigator (but not replace him) during his work by highlighting information to show him quickly the important information contained in the timeline and to ensure that he does not miss crucial information, in addition to provide explanations on these results. Subsequently, it is the responsibility of the investigation to transform and format these results to meet the specificities of the legal system of his country.

2.5/ CONCLUSION

This chapter has introduced digital forensics, the field of application of this thesis. Digital forensics is a complex field governed by strict rules inherent to the legal aspects of this area. During a digital investigation, an essential step is the reconstruction of past events. During the reconstruction, the investigators move from a static crime scene, containing footprints related to past activities, to a scenario describing the dynamics of the incident that led to the investigation. This scenario allows on the one hand to understand the circumstances of the incident and on the other hand to determine the responsibilities of the protagonists who have caused the incident. The scenario is also a source of evidence which, once admitted at trial, becomes proofs inculcating or exculcating the suspects.

As shown in Section 2.4, this process is however made complex by three problems: the volume of data, its heterogeneity and the legal requirements that have to be met to enable the production of evidence admissible at trial. Solving these three issues is a necessity to produce tools that can assist the investigators in the search for evidence, within a reasonable time and in compliance with legal requirements. A large number of event reconstruction approaches are proposed in the literature. Chapter 3 proposes a study of these existing solutions and identifies the strengths but also the limits of them to solve the mentioned issues.

As stated previously, the reconstruction of events is only one step of a digital investigation. Event reconstruction tools, to operate effectively, must be interfaced perfectly with the other steps of the whole process. Therefore, it is necessary to integrate our approach in this broader process. As said before, there is no consensus within the community on a definition of a digital investigation process. Consequently, the objective of Chapter 4 is to study exhaustively the existing digital investigation process models of the literature in

24 2. INTRODUCTION TO COMPUTER FORENSICS AND EVENT RECONSTRUCTION

order to identify the shortcomings of the existing and subsequently, create our own model for the integration of our approach.

EVENT RECONSTRUCTION

As said in Chapter 2, the reconstruction of events is an essential step in a digital investigation. The event reconstruction can be seen as a process taking as input a set of footprints and outputting a timeline of the events describing the case. The process of event reconstruction is complex due to three main issues which are the volume of data, the heterogeneity of data and the legal requirements that the results of the investigation have to meet to be admissible at trial. This chapter reviews the existing approaches for event reconstruction. For each of the three issues introduced in Chapter 2, a set of requirements is identified by the authors. We argue that these requirements must be met by every event reconstruction approach to allow the resolution of each issue. Subsequently, an approach that addresses the problems unsolved by the existing approaches will be presented in Chapter 5.

Event reconstruction approaches can be classified depending on the sources used and the time at which the tool is used. An event reconstruction tool can be based on a unique source (e.g. timestamp from file system) or based on multiple sources (e.g. logs files, file system, operating system information) (Inglo et al., 2012). In the first type of approach, the timeline does not fully represent what happened on the machine as all the footprints are not collected and therefore the investigators may miss important information. In the second type of approach (also called super-timeline approach), the timeline is more accurate than in the first approach, but the produced timeline is large and therefore difficult to analyse. A second distinction is made between tools that can use ex post evidence and tools that can use ex ante logging (Jeyaraman and Atallah, 2006). In the first case, the tool starts working after the incident happened and tries to identify and retrieve evidence to construct the timeline. In the second case, the tool starts working before the incident by recording all events occurring on the machine. When an incident occurs, the recorded information can be used to understand what happened. In this study, we focus on approaches which can cope with a large number of situations. Thus, we review only approaches that are able to work without prior knowledge of the systems studied during the investigation (ex post evidence approach). We present in this section ten approaches for event reconstruction. In addition and alongside the approaches that fully perform the

event reconstruction, this chapter introduces a number of relevant work to answer some related issues. This includes works on information traceability and event representation.

This chapter is organised as follows. Section 3.1 and the following three sections respectively introduce and evaluate the ten approaches studied in our state of the art. Section 3.2 assesses the abilities of approaches to answer three requirements related to data volume: the degree of process automation and the availability of analysis and visualisation tools. Section 3.3 studies the answers given by approaches to meet two requirements (the availability of mechanisms to deal with heterogeneous data sources and the completeness of the data representation model) related to heterogeneity. Section 3.4 evaluates the capacity of approaches to meet legal requirements. Two requirements are introduced in this evaluation: the ability of approaches to ensure the traceability of information and the level of credibility of the results produced. Finally, Section 3.5 is a discussion highlighting the strengths and the weaknesses of the proposed solutions.

3.1/ OVERVIEW OF EVENT RECONSTRUCTION APPROACHES

This section briefly presents the approaches that will be assessed in the rest of the chapter.

3.1.1/ ECF: EVENT CORRELATION FOR FORENSIC PURPOSES

In (Chen et al., 2003), the authors argue that it is possible to correlate the information contained in computers (log files, etc.) despite the heterogeneous nature of data. To reach this objective, an architecture named Event Correlation for Forensic Purposes (ECF) is proposed. This architecture is made of a database containing events extracted during an investigation in addition to tools able to instantiate and query this database. A set of ad-hoc extractors is used to complete the extraction of the information contained in the various sources that can be found in a crime scene. The system offers four main functionalities:

- Event extraction: this function enables to parse event sources, format events and populate the database. ECF proposes parsers to handle sources such as Apache logs, Windows 2000 logs or door logs.
- Dynamic queries: this interface enables the investigator to query the database. Queries are built by assembling constraints on one or more fields of the event table using Boolean operators. For example, the investigator may look for events occurring between two dates or search for all events caused by a given person.
- Custom queries: this interface enables to execute SQL queries. Therefore, this interface provides more flexibility to the user than dynamic queries.

- Hypotheses testing: this tool lets the user create new events and then, tests the validity of these assumptions.

3.1.2/ AUTO-ECF

In (Abbott et al., 2006), the authors proposed Automatic Event Correlation for Forensic Purposes (Auto-ECF) which is an evolution of the previous approach. Auto-ECF was designed to address several shortcomings of ECF. The main purpose of this approach is to provide automatic mechanisms to convert events extracted from heterogeneous sources to high-level events which are easier to understand for an investigator. To reach this objective, several concepts are introduced by the authors:

- Raw event: event contained in the event sources such as log files.
- Simple event: logical event resulting from the conversion of a raw event.
- Composite event: logical event resulting from the aggregation of several logical events.

To convert the raw events in logical events and to construct composite events, Event Logical Patterns (LEP) are used. After extracting the events from sources, a dedicated algorithm is used to search for occurrences of the patterns (stored in an XML file) and to create new events associated with each pattern.

3.1.3/ FORE: FORENSICS OF RICH EVENTS

The Forensics of Rich Events (FORE) approach proposed by (Schatz et al., 2004a) introduces a solution to deal with the large amount of data to be processed during an investigation and the difficulties encountered by investigators to interpret these data. FORE is an approach introducing a knowledge representation model (an ontology) for events occurring during an incident in addition to a set of extractors able to extract knowledge from sources such as Apache server logs and Windows 2000 logs. The FORE architecture is composed of two parts which are the extraction module and the analysis tools. The knowledge extracted is then used to populate the ontology with new instances of events. Each parser is dedicated to a specific type of source in order to take into account the specificity of each source. Regarding analysis, automated tools are proposed to process the knowledge stored in the ontology. A correlation tool based on rules is used to identify causal relationships between events. To express rules, a rule language called FR3 was created. A rule expressed with the language FR3 is made of antecedents and consequences. An inference engine is used to browse the knowledge base to find elements appearing in the antecedents of a rule. If all elements composing antecedents of a rule occur in the ontology, the rule is satisfied and elements appearing in the consequences of the rule are then added to the ontology.

3.1.4/ FINITE STATE MACHINE APPROACH

(Gladyshev and Patel, 2004) argues that a formalisation of the event reconstruction problem is needed to better structure the reconstruction process, facilitate its automation and ensure the completeness of the reconstruction. To address these problems, an approach based on finite state machine is proposed. The main idea of this proposal is to find the sequence of transitions that satisfies the constraints imposed by the evidences. First, the behaviour of the system under investigation is represented using a state machine. Subsequently, scenarii that do not match evidence collected by the investigators are removed. Once the number of potential scenarios has been reduced, a backtracking algorithm is used from the final state (the state observed at the beginning of the investigation) to the initial state of the system.

3.1.5/ ZEITLINE

Zeitline (Buchholz and Falk, 2005) is an editor enabling the investigator to create timelines from multiple sources of information. The interface of the tool enables the investigators to add new events to a given timeline, to aggregate several events to build a complex event or to search for specific events using a keyword-based query tool. This approach distinguishes two types of events: atomic events which are extracted from event sources and complex events containing several atomic or complex events. Whether they are complex or atomic, each event has a number of attributes including the date and time at which the event occurred, the name of the event, a description and a pointer to the parent event. In addition to these attributes, the atomic events and the complex events carry specific attributes such as the source of the event (for atomic events) and pointers to children for complex events. In addition to the possibility to extract events from various sources, users can create their own extractors enabling them to easily extend the number of event sources supported by Zeitline. The tool also offers to the investigators an interface enabling them to add new events to the timeline, to aggregate several events to build a complex event or to search for specific events using a query tool based on keywords. Finally, Zeitline is restricted by a number of rules that prevent the alteration of evidence. To prevent the modification of evidence, a system of views is also used to avoid the removal of information contained in evidence. When the investigator deletes an event from the timeline, the event is removed from a view but still physically preserved. This special attention given to the preservation of the integrity of the information is one of the main contribution of this tool.

3.1.6/ A FRAMEWORK FOR POST-EVENT TIMELINE RECONSTRUCTION USING NEURAL NETWORKS

On the basis that most of the existing methods cannot efficiently handle large volumes of data, (Khan et al., 2007) introduces a new approach using a neural network. The main motivation of this work is to show the ability of machine learning techniques to process large datasets. This approach uses traces left by user activities in the system to detect the activity of software. The proposed tool is made of three parts: the parsers that extract traces found in various types of sources (log files, registry, etc.), the preprocessor used to convert data extracted by parsers to make it usable by the neural network and the neural network used to identify launched applications using input data. The use of machine learning techniques also explicit the reasoning made to produce a conclusion which is one of the justice requirements.

3.1.7/ FACE: FORENSICS AUTOMATED CORRELATION ENGINE

(Case et al., 2008) point out that the consultation of data produced during an investigation is a tedious work. As the current forensic tools are limited to the extraction and presentation of information extracted from sources, there is an important need to develop tools able to assist investigators during the interpretation and the analysis of the data. In this work, an approach, called Forensics Automated Correlation Engine (FACE), focusing on the interpretation and the analysis of the data is proposed. FACE is able to handle five different data sources which are memory dumps, network activities, disk images, log files and user configuration files. Once data is extracted, the correlation tool discovers relationships between events and between objects and events (e.g. a file). At the end of the process, the investigators get a report describing the activities of the user (events, objects and logical relationships between events and between events and objects). This report is made of activities linked by hyperlinks to facilitate the consultation of the timeline.

3.1.8/ CFTL: CYBER-FORENSIC TIME LAB

(Olsson and Boldt, 2009) discusses the need for a system to view and navigate into the data related to an investigation in an intuitive way to discover evidence more easily. To reach this objective, the tool, called Cyber-Forensic TimeLab, described in this work extracts timestamps found in a machine or a group of machines, builds the timeline and then provides a graphical view of all the events. The investigator can then browse the events and identify relevant information more easily. The proposed tool is made of two parts: a scanner and an event viewer. The scanner is used to extract timestamps from sources (file system, Windows or Unix logs, JPEG files) and store them in an XML file. Each evidence has three required attributes (name, type and identifier) and several op-

tional attributes. Once timestamps are extracted, the event viewer reads the XML file, orders events and then displays them in a graphical timeline. The main added value of this approach is the improvement of the ergonomics of the interface between the timeline and the investigator.

3.1.9/ PLASO

Plaso¹ is a toolkit constructing automatically super-timelines using a large number of sources (Windows event logs, web browser histories, Apache logs, PDF metadata, fire-wall logs, etc.) from a disk image. Plaso is the solution that handles the largest number of information sources in the literature. To reach its goals, it is made of several tools including *log2timeline* and *psort*. *log2timeline* is a tool proposed by (Gudhjonsson, 2010). It is the main component of Plaso as it makes it possible to build the super-timelines. In addition to *log2timeline*, Plaso includes a tool called *psort* that can be used to format the result produced by *log2timeline* as a text file, a CSV file, a database, etc. The default output format is composed of a limited number of fields to store the date on which an event occurred, the source that has been used for the extraction of the event and a message describing the event.

3.1.10/ PYDFT

Python Digital Forensic Timeline (PyDFT) (Hargreaves and Patterson, 2012) is proposed as part of the development of a system reconstructing high-level events (which are easier to understand for investigator) from low-level events (which are extracted from information sources by *log2timeline* or *Zeitline*) in order to improve readability of the final timeline. (Hargreaves and Patterson, 2012) states that the number of events generated by super-timeline approaches make the visualisation and therefore the analysis of the timeline complex. The authors also try to meet the needs of justice by storing traceability information during the process of summarisation. For each high-level event, the investigator has therefore the possibility to know the low-level events used to create it. The proposed solution implements a two-step process: the extraction of low-level events and the construction of high-level events. A system made of parsers and bridges is used to carry out the low-level event extraction. Parsers are used to process the content of sources. Two types of sources are used: the file system and the information contained in the files themselves. Then, bridges convert the extracted data into the format used for low-level events. Once the low-level timeline is built, patterns are searched in it and corresponding high-level events are added accordingly. The summarisation of a timeline is a relevant functionality as it facilitates the reading of the timeline and by extension, its analysis.

¹<http://plaso.kiddaland.net/>

In this section, ten event reconstruction approaches have been introduced. After having explain the functionalities proposed by each of them, the next section evaluates the solutions provided by each approach to face the three issues of the event reconstruction: the data volume to handle, the heterogeneity of this data and the legal requirements.

3.2/ DATA VOLUME

As said in Chapter 2, one of the main challenge during the event reconstruction is the manipulation of large volumes of data. This challenge raises three problems that approaches have to solve in order to be able to deal with large data volumes:

- **Automated construction of timelines** (see "Automation" in Table 3.1): the volume of data hinders the ability of investigator to build manually a complete timeline from the footprints found in the crime scene. Indeed, the construction of a timeline involves handling and integrating a lot of information from different sources. Therefore, the provision of tools to build automatically the timeline is a crucial aspect to enable efficient processing of the information.
- **Analysis of information** (see "Analysis" in Table 3.1): the volumes of data make complex the understanding of it and the manual identification of links between pieces of data (is an event linked to another and if so, how and why?). Get a comprehensive understanding of a timeline is a tedious task that lead in most cases in cognitive overload. For these reasons, we argue that another requirement for the approaches is to make available tools helping the investigator to understand the incident and to draw conclusions. The aim of analysis tools is to highlight relevant information to give research directions to investigators. This includes making the timeline easier to read by filtering data or summarise the timeline as proposed in (Abbott et al., 2006) and (Hargreaves and Patterson, 2012), identify correlations between events as in (Schatz et al., 2004a) and (Case et al., 2008) and produce conclusions from the knowledge contained in the timeline.
- **Visualisation tools** (see "Visualisation" in Table 3.1): the large volumes of data make the consultation of information very difficult for users. This causes significant cognitive overload for investigators and delay in their work. Moreover, this can also lead them to miss important information. Therefore, based on (Olsson and Boldt, 2009), we state that the last requirement for event reconstruction approaches regarding the challenge of data volume is the provision of visualisation tools enabling to browse data in an efficient, clear and intuitive way. In addition to the availability of data for automatic processes, the model must also enable investigators to access and understand the information easily. In particular, models should enable the use of search and visualisation tools to make data available to investigators in an understandable and intuitive form.

In the rest of this section, we evaluate the existing solutions of event reconstruction in the light of these requirements.

3.2.1/ AUTOMATED TIMELINE GENERATION

In the literature, a large number of approaches provide tools to automatically extract the information and populate a central storage constituting the timeline. In ECF (Chen et al., 2003), an architecture based on a database containing events extracted during an investigation. In addition, it introduces a set of extractors to collect events and store them in a database, which makes it possible to quickly generate a temporal ordered sequence of events. These automatic extractors, a widely used concept, can also generate the timeline as in FORE (Schatz et al., 2004a), FACE (Case et al., 2008), CyberForensic TimeLab (Olsson and Boldt, 2009), Plaso and PyDFT (Hargreaves and Patterson, 2012). However, in some approaches, including (Gladyshev and Patel, 2004) and (James et al., 2010), the lack of automation does not allow to handle complex cases. Indeed, the investigation of a single computer may involve several processes such as web browsers, file system, instant messaging software, etc. Thus, the representation of such a system with a finite state machine seems not possible. Second, the use of finite state machine leads to combinatorial explosion when used on real cases. (James et al., 2010) proposes to convert the finite state machine into a deterministic finite state machine to limit the exponential growth of the size of the machine and therefore the number of scenarios to examine during the backtracking algorithm. Despite the reduction of the size of the state machine, the experiments show that the approach can still not be used on real forensic cases.

3.2.2/ ANALYSIS TOOLS

The automated extraction from a large number of sources lead to the creation of large timeline that are difficult to read, interpret and analyse for the investigators. To assist the investigators during this phase, event reconstruction approaches have to provide analysis tools carrying out all or part of the reasoning.

Among analysis tools found in the literature, there are several proposals of tools to find correlations between events and tools enabling to summarise timelines. Correlations tools, as proposed in (Schatz et al., 2004a) and (Case et al., 2008), make it possible to identify relationships between events such as causal relationships. Despite the relevance of this tool, the existing proposals are based on rules. A rule-based system requires the user to define the set of rules which is a tedious task. In addition, this set of rules is bounded and thus, it cannot take into account all cases. Indeed, the completeness of the set is a particularly difficult goal to achieve and it cannot take into account cases that are still unknown. In addition, using a set of rules requires to update it regularly. Event reconstruction approaches must implement algorithms that can adapt to any kind

of situation even those unknown for investigators. It is, therefore, necessary to develop analysis tools not relying only on base of rules defined by the user.

The summarisation of a timeline is an analysis tool enabling to reduce the volume of data to handle by the investigators by making the visualisation of it clearer. It is a relevant functionality as it facilitates the reading of the timeline and by extension, its analysis. In the existing works, there are several proposals of summarisation techniques including (Abbott et al., 2006) and (Hargreaves and Patterson, 2012). Both in Auto-ECF and PyDFT, the authors distinguish between the raw events (event that are contained in the data sources such as log files and that are directly extracted from the crime scene) and the composite events (logical event resulting from the aggregation of several raw events). Pattern-based tools are used to browse the timeline made of raw events to discover sequences corresponding to a given composite event. Therefore, the summarisation has the same issues that the correlation tools. The use of pattern-based techniques does not allow to build flexible tools and the definition of patterns remain the responsibility of the user.

Other works propose analysis tools to answer specific questions not directly related to the correlation or the summarisation of events. This is the case of (Khan et al., 2007) where a neural network-based approach using footprints left in a machine to detect software activities is introduced. Unfortunately, the performance of the proposed tool is poor. The training of the neural network and the need to use it several times to get a complete scenario make the use of this tool time-consuming. Finally, most of the tools do not provide analysis capabilities as for Zeitline for which the investigators have to handle themselves the aggregation of events to create high-level events.

3.2.3/ VISUALISATION TOOLS

Approaches that are able to handle large volume of data have to be associated with visualisation tools presenting this large amount of data in a clear and intuitive way. Some approaches offer ergonomic ways to access information like (Case et al., 2008) where reports, describing the entities and relationships between entities, are generated by the system. Zeitline (Buchholz and Falk, 2005) provides a graphical editor enabling the investigator to create timelines from multiple sources of information. The tool does not offer automatic analysis process, but invites the investigators to handle themselves the aggregation of events to create high-level events. Zeitline integrates a graphical interface to browse the timeline. In (Case et al., 2008), the proposed tool generates a report that offers different views on events and objects in addition to hyperlinks between them to make the timeline easier to read and more intuitive for investigators. In (Olsson and Boldt, 2009), a system called Cyber-Forensic TimeLab is proposed. The focus of this work is the conception of a system to view and navigate into the data related to an investigation in an intuitive way to discover evidence more easily. Then, the timeline is displayed in a graphical browser that is the main added value of this approach as it constitutes an

improvement of the ergonomics. Finally, in (Schatz et al., 2004a), an event browser is proposed to interact with the events contained in the knowledge base. This browser includes two different views: the event causality view and the entity view. The first displays events and their causal ancestors while the second shows all entities contained in the base in addition to their properties. The user can search specific events and entities in those two views using a query interface based on event types and property values.

Many tools do not offer an intuitive interface, but only a query tool that appears to be a powerful but complex and tedious way to access the information. This is the case of approaches using a database and providing a SQL query interface which is efficient but not intuitive for untrained users as in (Chen et al., 2003) where two SQL-based query systems (i.e. dynamic and custom queries) are proposed or as in (Hargreaves and Patterson, 2012).

3.2.4/ SYNTHESIS

The Table 3.1 shows a comparison of existing approaches with regard to the three requirements studied in this section (their strengths (✓), limitations (✗), partial or inadequate solutions (●)).

Approach \ Criterion	Automation	Analysis	Visualisation
ECF (Chen et al., 2003)	✓	✗	✗
Auto-ECF (Abbott et al., 2006)	✓	●	✓
FORE (Schatz et al., 2004a)	✓	●	✓
Finite State Machine (Gladyshev and Patel, 2004)	✗	●	✗
Zeitline (Buchholz and Falk, 2005)	✓	✗	✓
Neural networks (Khan et al., 2007)	●	✗	✗
FACE (Case et al., 2008)	✓	✗	✗
CyberForensic TimeLab (Olsson and Boldt, 2009)	✓	✗	✓
Plaso (Gudhjonsson, 2010)	✓	✗	✗
PyDFT (Hargreaves and Patterson, 2012)	✓	●	✓

Table 3.1: Evaluation of event reconstruction approaches regarding the problems related to data volume

3.3/ HETEROGENEITY

Because the footprints are spread across the crime scene in many different sources of information (i.e. logs, file system, etc.), the investigators have to face heterogeneity problems. We can classify the heterogeneity inherent to event reconstruction into three categories. First, the **syntactic heterogeneity** is due to the information encoding is not the same among sources due to the formatting. Therefore, depending on the source, footprint data may be different. It is therefore necessary to know the context of data to determine its meaning. The context of a particular footprint is composed of the file format or the program used to create it for example. Second, the **semantic heterogeneity**

depicts the fact that a same event can be interpreted or represented in different ways. For example, an event describing the visit of a webpage may appear in different ways in web browser logs and server logs. Third, the **temporal heterogeneity** is due to the use of different sources from different machines may cause timing problems. First, there are some issues due to the use of different time zones and synchronised clocks. Second, the temporal heterogeneity can be due to the use of different formats or granularities (e.g. 2 seconds in FAT file systems, 100 nanoseconds in NTFS file systems). In our research, we focus on the first two types of heterogeneity as they are both related to our desire to reduce the cognitive overload by providing to investigators a clear and complete view on the past events. To solve the problems of heterogeneity, an approach must meet the following points:

- **Management of multiple sources** (see "Multiple Sources" in Table 3.2): Due to the semantic and syntactic heterogeneity of data, it is very hard to get a comprehensive view of the incident. To reach this objective, it is necessary to handle all the information contained in the various sources that can be found in the crime scene (Gudhjonsson, 2010), without semantic losses during the integration of the information into a global model. Therefore, we argue that the first requirement for an approach of event reconstruction is the implementation of mechanisms (e.g. parsers) to process multiple and various footprint sources and to federate the information collected in a coherent and structured way.
- **Completeness of the data model** (see "Completeness" in Table 3.2): To guarantee the re-usability of the information collected in the crime scene, during all the investigation process, it is necessary to introduce a global model which represents all the various and heterogeneous types of data without semantic losses (Chabot et al., 2015b). The proposed model must be complete enough to represent accurately the events that occur during an incident. To do this, the model must provide a vocabulary sufficiently developed to model the entities related to a digital incident, their characteristics and the relationships between them.

In the rest of this section, we evaluate the existing solutions of event reconstruction in the light of these two requirements. The handling of heterogeneous sources requires on the one hand the development of automated extractors that can extract information specific to each source and on the other hand a sufficiently complete knowledge model to represent all aspects of a digital incident.

3.3.1/ MANAGEMENT OF MULTIPLE SOURCES

(Gudhjonsson, 2010) states that managing an insufficient number of sources makes the truthfulness of the timeline vulnerable to anti-forensics techniques (e.g. alteration of timestamp). In addition, the quality of the timeline also suffers from the small number of sources. For example, some contextual events may not appear in the timeline. To en-

hance the quality of the timeline and to minimise the impact of anti-forensics techniques, the author argues that event reconstruction approaches must handle a large number of sources. To validate this idea, (Gudhjonsson, 2010) has proposed the Plaso toolbox which is the tool that handles the largest number of information sources. Plaso makes possible the construction of super-timeline (timeline integrating many sources of events) from a wide range of sources including file system logs, recycle bin, registry etc.

A large part of existing approaches follow this idea and are able to deal with multiple and heterogeneous sources including Windows event logs, web browser histories, Apache server logs, files meta data, instant messaging software, registry, memory dumps, network activities and user configuration files. The commonly used solution is the implementation of extractors dedicated to each source of information to identify and extract relevant information and populate a data structure with that knowledge. However, some approaches, such as (Gladyshev and Patel, 2004) and (Khan et al., 2007), suffer from the inclusion of an insufficient number of sources which can lead to a loss of relevant information.

3.3.2/ KNOWLEDGE REPRESENTATION FOR EVENT RECONSTRUCTION

Most of event reconstruction approaches used a data structure to store the knowledge gathered during the investigation. This structure, which is more or less complex depending on the chosen solution, is the central element of an approach and therefore has an impact on the strengths and weaknesses of it. The data model makes it possible to structure the data and to standardise its representation. The choice of the model affects three aspects of an event reconstruction approach. The completeness and the accuracy of the model first determines the capacity of the approach to faithfully represent the information extracted from the crime scene. The capacity and the quality of analysis and visualisation tools are also directly impacted by the choice of the data model. Indeed, the degree of structuring of the data model facilitates or not the implementation of these tools.

This subsection studies different knowledge representation models proposed in the literature to identify the most appropriate structure to meet the needs of event reconstruction. We also present models that are not integrated into a comprehensive approach of event reconstruction, but that could however provide the basis for such an approach. We distinguish two types of data structure: data formats and advanced data models. The data formats, as defined in (Carbone and Bean, 2011) are textual timelines that are difficult to read and understand. We can distinguish three levels of data formats for timelines: the preliminary timeline formats, the intermediate timeline formats and the final timeline formats. The preliminary timeline formats are very difficult to read and understand as they are closer to logs produced by software while intermediate and final timeline formats are considered as formats that can be directly used by the investigators, although it is recommended to use transformation processes to make the timeline more useful and

understandable (Carbone and Bean, 2011). For their part, the data models are data structures that are more complex and structured than data formats. Databases and ontologies are two examples of data models.

3.3.2.1/ DATA FORMATS

Bodyfile (Farmer and Venema, 2004) is a data format for representing textual timelines of events using few attributes. Bodyfile is a preliminary data format proposed as part of the development of the Coroner's Toolkit. This format is used to store information about objects (files) detected in a system. Bodyfile is composed of eleven fields, including the name of the object and its identifier, its size, information about the user related to the object and the dates of last access, last modification of the object or its metadata and creation of the object. Timeline (Carvey, 2009) is an intermediate timeline format storing information about events that have occurred on machines. This format is made of five fields: the date and time of the event and a description of it, the information source used to identify it, a description of the host system and an identifier of the user related to the event. Among tools proposed in Plaso, *log2timeline* extracts events from a disk image and *psort* can be used to format the result produced by *log2timeline* as a text file, a CSV file, a database, etc. The text format is made of a limited number of fields to store the date on which an event occurred, the source that has been used for the extraction of the event and a message describing the event.

Using a small number of features offers higher performance than more complex models. However, the features introduced in these data formats do not enable to accurately represent any event occurring on a machine. Nevertheless, increasing the number of attributes is not a satisfactory solution as it makes heavy format and thus reduces readability. Moreover, the use of a text format does not allow to explicitly and simply show the relationships between entities. The data formats are therefore a low-tech solution that offers interesting performances for simple use cases, but which is limited in terms of expressiveness for complex use cases. Another limitation is that the textual formats are particularly difficult to handle for humans (this is especially true for preliminary timeline formats and intermediate timeline format). The textual format is not a relevant medium of communication to allow investigators to quickly and easily understand the information in the timeline.

To address these issues, several XML-based data formats are proposed in the literature. In the Cyber Observable Expression (CybOX) project (Barnum, 2011), a set of XSD schemas is introduced to represent any entity (process or object, i.e. cyber-observables) observed during an incident in addition to the interactions between the cyber-observables and events affecting them. The development of these schemas starts from the observation that each domain and each tool currently use its own representation. This undermines the coherence, the effectiveness and the interoperability of the tools. CybOX en-

ables the representation and the sharing of events. It is based on two schemes which define the structure and the features of it: *cybox_core* and *cybox_common*. *cybox_core* defines the structure of CybOX which is composed of six main concepts: Action, Event, Object, Observable, Observables and Property. The use of these six concepts allows to describe accurately all the events that can occur on a machine. In addition to these general notions, objects are defined in individual XSD files. The proposed schemas integrates knowledge from experts through a system of objects specialising the abstract notion of object. The large set of objects provided offers the ability to represent a PDF file, an HTTP session, a web history, a Windows process, a network connection, etc. Although CybOX is very complete, it can be improved by implementing it using other technologies than XSD/XML. XSD/XML is a serialisation format and therefore, it does not allow to model the semantics of the data. The unavailability of the meaning of data for the algorithms consequently reduces their ability to analyse it. This limitation is one of the motivations of the introduction of more complex data structure such as ontology.

3.3.2.2/ DATA MODELS

ECF (Chen et al., 2003) stores data in a database consisting of a table containing common information about events and tables containing information specific to each type of event. One of the objectives is to provide a canonical form for representing events uniformly regardless of the source from which they come. Adopting a generic information level and a specialised information level is an idea also used in CybOX (Barnum, 2011). This conceptual separation introduces a canonical representation of events that facilitates information processing (analysis tasks for example) while preserving the specificities of each event. For this, a set of characteristics common to any events occurring on a machine. This set includes an identifier for the event, the date and time at which it occurs, information about the actor who caused the event (e.g. IP address), information about the object affected by the event (URL if the object is a webpage for example), the action represented by the event, the result of the event (success, failure, unknown) and information about the source used to identify the event. A second table is used to store specific information about events depending on the source from which they are extracted. However, no details are given by the authors about this second table. Despite the relevance of the separation into two tables, the use of a database does not allow to take full advantage of this feature. Unlike ontology, databases do not allow to explicitly represent the semantics of data which constrains the understanding of data by analysis algorithms. Thus, a large part of the investigation has to be carried out by investigators. In addition and as in the FORE approach (Schatz et al., 2004a), the ECF model does not define relationships between entities (subject, object, event). For example, it enables to model the fact that an event interacts with an object, but the nature of this interaction cannot be defined accurately.

The FORE ontology (Schatz et al., 2004a) consists of classes that represent the notion of Entity (i.e. objects composing the world) and the notion of Event (i.e. the change of state of an object over time). The Event entity may itself be inherited by other classes in order to describe different types of events that can occur on a machine. Events can be linked using an object property representing causality (event A causes event B if event A has to happen before event B to allow it to occur). A number of attributes are used by classes inheriting from the Event class to provide information about the user or the process that produced the event, the files used by it, etc. The use of an ontology to represent events is an efficient way to deal with heterogeneity issues and allows the introduction of semantically rich models able to capture the semantics of all entities and the relationships linking them. The precise and formal description of the components of the model can significantly increase analysis capabilities by enabling machines to understand the meaning of the data (in contrast to unstructured formats). Unlike textual formats, ontologies can provide better data visualisation using graphs that enable the user to easily view the connections between entities.

3.3.3/ SYNTHESIS

Approach \ Criterion	Multiple sources	Completeness
ECF (Chen et al., 2003)	✓	●
Auto-ECF(Abbott et al., 2006)	✓	●
FORE (Schatz et al., 2004a)	✓	●
Finite State Machine (Gladyshev and Patel, 2004)	✗	Not applicable
Zeitline (Buchholz and Falk, 2005)	✓	✗
Neural networks (Khan et al., 2007)	●	Not applicable
FACE (Case et al., 2008)	✓	Not applicable
CyberForensic TimeLab (Olsson and Boldt, 2009)	✓	✗
Plaso (Gudhjonsson, 2010)	✓	✗
PyDFT (Hargreaves and Patterson, 2012)	✓	✗

Table 3.2: Evaluation of event reconstruction approaches regarding the problems related to heterogeneity

The Table 3.2 shows a comparison of existing approaches with regard to the requirements studied in this section (strengths (✓), limitations (✗), partial or inadequate solutions (●) or "not applicable").

3.4/ LEGAL REQUIREMENTS

The legal requirements are an important aspect of every digital investigation as their compliance ensures the admissibility of results at trial. As emphasises in Chapter 2, the evidence produced in a court must meet several criteria, including credibility and reproducibility of the digital evidence (Baryamureeba and Tushabe, 2004). In the rest of this section, we evaluate the existing solutions of event reconstruction in the light of these

two requirements. It appears that these two criteria are linked. Indeed, the credibility of results is directly affected by the traceability of the information (i.e. the capacity to explain the results produced by the tools). Therefore, we choose, in the following study of existing approaches, to consider that these two requirements are only one requirement covering both aspects of credibility and reproducibility (see "Traceability and Credibility" in Table 3.3).

The **reproducibility** allows a court to fully understand the path (made of data manipulation steps, reasoning steps, etc.) used to reach each conclusion which is presented during a trial and to evaluate the quality of this process. Thus, a model must allow the integration of information on the provenance of data produced during the investigation. This includes storing each step of an investigation, the investigators involved in activities, the tools or the techniques used to produce the information (e.g. extraction from a data source, deduction from two pieces of information already known, etc. The level of **credibility** of evidence is directly related to the level of accuracy and verifiability of the method and the information source used to produce the evidence. By detailing exhaustively the investigation process, the modelling of the provenance of information can increase significantly the credibility of all evidence produced by the investigative process. Include information about provenance in a model has many benefits:

- Trace activities that alter the state of an information at various stages of an investigation, from the collection phase in the crime scene to the presentation of results.
- Help to determine the level of trust of information.
- Ensure that the tools used to produce a result respect the rules established by the laws.
- Facilitate the understanding of a result by memorising how information is derived from other information and what are the evidence used to support the process of reasoning.
- Allow the reproduction of the investigative process by memorising all the steps which were used to reach a conclusion.

In addition, to give credibility to the results produced by the investigation, a model based on a sound and formal theory is needed.

Few approaches are able to explain how the results are obtained. Traceability is particularly lacking in (Khan et al., 2007). Unlike other machine learning techniques, the use of neural networks does not allow to explicit the reasoning used to produce results (which is one of the justice requirements) as they can be considered as black boxes. Indeed, some settings used during the learning phase remain unknown. The model proposed in (Hargreaves and Patterson, 2012) stores low-level and high-level events, and is composed of two structures to represent them. The first one includes attributes to model the date and the type of the event and the source used to identify it. The second one has a similar structure in addition to attributes to memorise how each high-level event is generated

using low-level events. This enables to store information about data provenance.

In addition to previous solutions, the field of research related to data provenance gives answers to the problems of traceability and reproducibility. As defined by (Gil et al., 2010), the provenance of an object or a piece of information is a record describing the entities and processes involved in its creation, dispersion or other activities that affect the object. The notion of provenance provides a fundamental basis to evaluate the authenticity and the truthfulness of a resource. Assertions about provenance take the form of metadata for which provenance can itself be described. The most significant contribution of this research field is the ontology Provenance Ontology (PROV-O) (Lebo et al., 2013), a W3C recommendation of the representation of the provenance of information. This ontology is composed of concepts and relationships, allowing to define a piece of information, to assign this information to a user or an entity and to represent the process used to produce the information. Despite of its quality, this ontology is not directly applied to the field of computer forensics and therefore, the proposed model is too generic to answer the need of the digital forensics field. Thus, to be usable, this ontology needs to be adapted.

CybOX (Barnum, 2011) also incorporates elements to model the provenance of information enabling to memorise, for each cyber-observable, the source of information and the techniques (extraction, analysis, etc.) used, the contributors who helped to produce the cyber-observable and tools used. It can also represent information about the noise affecting the extraction of information and the level of difficulty to obfuscate a cyber-observable. This information can then be used to evaluate the truthfulness of the information. In addition, CybOX is enriched by Digital Forensic Analysis eXpression (DFAX) (Casey et al., 2015), an ontology representing information about the provenance of information. DFAX provides an extra layer to represent forensic actions initiated by the investigators.

Digital Forensics Extensible Markup Language (DFXML) (Garfinkel, 2012) is an XML language for the representation of forensic data. It makes it possible to represent a large range of information, including files, forensic tools used to process the information, the state of the computer studied, the evidence extracted, etc. The aim of this language is to facilitate the sharing of information between the forensic tools and the investigators.

Concerning the credibility of results, only (Gladyshev and Patel, 2004) and (Khan et al., 2007) are supported by recognised theories (respectively, finite state machine and neural networks). As a prelude to their work, (Gladyshev and Patel, 2004) argued that a formalisation of the event reconstruction problem is needed to simplify the automation of the process and to ensure the completeness of the reconstruction. The use of finite state machine, a well-known theory in the scientific community, gives strong theoretical foundations to this approach. However, as for (Khan et al., 2007), this approach suffers from other limitations such as poor performances and inability to handle large data volumes.

The Table 3.3 shows a comparison of existing approaches with regard to the requirement studied in this section (strengths (✓), limitations (✗), partial or inadequate solutions (●))

or "not applicable").

Approach \ Criterion	Traceability and Credibility
ECF (Chen et al., 2003)	Not applicable
Auto-ECF (Abbott et al., 2006)	●
FORE (Schatz et al., 2004a)	×
Finite State Machine (Gladyshev and Patel, 2004)	●
Zeitline (Buchholz and Falk, 2005)	●
Neural networks (Khan et al., 2007)	●
FACE (Case et al., 2008)	×
CyberForensic TimeLab (Olsson and Boldt, 2009)	×
Plaso (Gudhjonsson, 2010)	×
PyDFT (Hargreaves and Patterson, 2012)	●

Table 3.3: Evaluation of event reconstruction approaches regarding the compliance with the legal requirements

3.5/ DISCUSSION

This chapter highlights the strengths and the limitations of ten approaches of event reconstruction. We have especially studied the solutions proposed to address the challenges related to the three main issues of event reconstruction identified in Chapter 2. First, regarding the large volumes of data to handle, the extraction of information from large and heterogeneous sources appears to be a solved problem. Thanks to solutions such as the Plaso toolbox (Gudhjonsson, 2010), it is possible to build automatically and efficiently timelines depicting the events that have occurred in the past. However, the assistance to investigators during the analysis remains a limitation for most approaches. Two cases can be distinguished in the literature. The first case is the approaches with reasoning abilities restricted by the choice of approach (e.g. the approaches based on finite state machine face a combinatorial explosion on real cases) or the choice of the data structure used to store the knowledge (e.g. use of textual formats or a database for example). The second case is the approaches based on ontology or other data structures adapted for reasoning which take advantage of the inherent qualities of this structure. This is especially true for approaches such as FORE (Schatz et al., 2004a) and PyDFT (Hargreaves and Patterson, 2012). In the case of ontology-based approaches, the use of an ontology makes explicit the semantics of the data and therefore enables the use of advanced analysis tools. However, the proposed tools (i.e. correlations tools and summarisation of timelines) are based on sets of rules defined by the users. The construction of such a set is a tedious and time consuming task due to the large number of rules that must be defined and the need for regular updates. In addition, such methods do not allow to take into account all the cases, the completeness being a very difficult goal to achieve. But to be effective, the analysis tools need to be able to produce results, even for unknown cases. Thus, the approaches must offer more generic analysis tools and therefore tools not only based on rules defined by the user. Therefore, there is a need for analysis tools

that can adapt to various types of situations, even unknown.

Similar to the handling of large volumes of data, the extraction of information from heterogeneous sources is a problem solved by several approaches including Plaso. However, as shown in Section 3.3, the models used to store data are not comprehensive enough to store all the information that can be collected from a crime scene. As a consequence, the models do not allow to represent accurately and faithfully a digital incident. Information about past events are therefore lost and this reduces the analysis capabilities.

Regarding the legal requirements, as said in Section 3.4, few approaches propose mechanisms to clearly explain the intellectual process used to produce the results. In addition, some approaches are based on well-know and recognised theories such as (Gladyshev and Patel, 2004) and (Khan et al., 2007). However, these approaches suffer from their poor performance, which does not allow them to operate efficiently on real cases.

DIGITAL INVESTIGATION PROCESS MODEL

The field of computer forensics is an area governed by strong legal constraints and requirements. Compliance with these rules ensures that the tools are able to produce evidences that are admissible in a court. A digital investigation follows a precise process defined upstream the investigation. This process must help to achieve the objectives of the investigation by ensuring compliance with the legal requirements. As said in Chapter 2, the event reconstruction is one of the numerous steps composing an investigation. As our work is a proposal of an approach for the reconstruction of events, it is necessary to properly interface it with the wider process that is the investigation. However, there is no clear consensus currently on a process model for digital investigations. This chapter studies the strengths and weaknesses of existing investigative process models to guide us in creating our own process model to integrate our approach.

This chapter is structured as follows. Section 4.1 reminds the interests of the definition of such a model for the resolution of the problems (data volume, heterogeneity, legal requirements) identified in Chapter 2. Then, Section 4.2 proposes an exhaustive state of the art of existing models. Finally, Section 4.3 is a discussion to highlight the strengths and weaknesses of existing models and to set the foundations for the creation of our model.

4.1/ INVESTIGATION PROCESS MODEL: NEEDS AND CHALLENGES

The creation of a comprehensive process model for digital investigations pursues several objectives. First of all, the main purpose of such a model is to help investigators to meet legal requirements. The definition of an investigation process model can contribute to the completion of these prerequisites in several ways:

- As said in the previous chapter, the level of **credibility** of an evidence is directly related to the accuracy and the verifiability of the methods used to produce it. By detailing exhaustively the investigation process, a process model can increase significantly the credibility of all evidences produced by the investigative process. Making available the investigative process through an accurate and consistent model enables the court to evaluate the quality of this process. In addition, the confidence of the court in the process can be influenced by a broad consensus among the digital forensics community on the model.
- The **reproducibility** of an evidence expresses the ability to reproduce the process allowing to produce an evidence. The establishment of a well-defined investigation process model contributes to the reproducibility of results by exhibiting steps by which the investigators have gone to reach the results of the investigation.

The second purpose of making available a comprehensive investigation process model is to provide a framework for the development of digital forensic software tools (Ciardhuáin, 2004), (Reith et al., 2002). The size of the data to handle during a digital investigation motivates the development of tools able to carry out automatically part of the investigative process. To guide the development of digital forensics software tools, a specific and highly detailed process model is needed to allow an easy translation into algorithms. This framework should include a definition of the steps composing a digital investigation including: 1) the preservation of the crime scene, 2) the definition of the crime scene, 3) the collection of footprints, 4) the construction and the analysis of the timeline and 5) the presentation of conclusions to Justice. During the development of this process model, the focus should be on the precision and completeness of the description of the meaning of each step and the definition of data flow entering and leaving each step.

4.2/ DIGITAL INVESTIGATION PROCESS MODEL: A STATE OF THE ART

In this complete and exhaustive state of the art, we studied eighteen process models which are listed in Table 4.1. This section provides an overview of the completeness and coverage of each model by listing the steps composing them. It should be noted that investigation steps identified in this section are the result of a pre-selection of the most common tasks found in these process models. Therefore, the steps that are less represented are not taken into account in this study, but are described in section 4.2.4. In addition, some steps may have been grouped under a single step or renamed.

An investigation can be decomposed in three main phases which are the pre-investigation, the investigation (called investigation core in this chapter) and the post-investigation. The nature and the goals of each phase are defined below.

ID	Investigation process model
CFIP	Computer Forensic Investigative Process (Pollitt, 1995)
DFRWS	DFRWS Investigation Model (Palmer, 2001)
ADFM	Abstract Digital Forensic Model (Reith et al., 2002)
IDIP	Integrated Digital Investigation Process (Carrier et al., 2003a)
EEDI	End to End Digital Investigation (Stephenson, 2003)
EDFIF	Event-based Digital Forensic Investigation Framework (Carrier and Spafford, 2004b)
EIDIP	Enhanced Integrated Digital Investigation Process (Baryamureeba and Tushabe, 2004)
EMCI	Extended Model of Cybercrime Investigation (Ciardhuáin, 2004)
HOFDIP	An Objective-based Framework for the Digital Investigation Process (Beebe and Clark, 2005)
CFFTPM	Computer Forensics Field Triage Process Model (Rogers et al., 2006)
FDFI	Framework for a Digital Forensic Investigation (Kohn et al., 2006)
CPMICF	Common Process Model for Incident and Computer Forensics (Freiling and Schwittay, 2007)
DFIPM	New Digital Forensics Investigation Procedure Model (Shin, 2008)
DFMMIP	Digital Forensic Model based on Malaysian Investigation Process (Perumal, 2009)
GCFIM	Generic Computer Forensic Investigation Model (Yusoff et al., 2011)
SDFIM	Systematic Digital Forensic Investigation Model (Agarwal et al., 2011)
PFP	Proactive Forensic Process (Alharbi et al., 2011)
MHEI	Process model for the Automation of Digital INvestigations (Vlachopoulos et al., 2013)

Table 4.1: The eighteen process models studied in our state of the art

4.2.1/ PRE-INVESTIGATION

ID	Operations readiness	Infrastructure readiness	Detection & Alert	Confirmation	Authorisation	Preparation	Notification
CFIP	✗	✗	✗	✗	✗	✗	✗
DFRWS	✗	✗	✓	✗	✗	✗	✗
ADFM	✗	✗	✓	✗	✓	✓	✗
IDIP	✓	✓	✓	✗	✓	✗	✓
EEDI	✗	✗	✗	✗	✗	✗	✗
EDFIF	✓	✓	✓	✗	✓	✗	✓
EIDIP	✓	✓	✓	✗	✓	✗	✓
EMCI	✗	✗	✓	✗	✓	✗	✓
HOFDIP	✓	✓	✓	✗	✗	✗	✗
CFFTPM	✗	✗	✗	✗	✗	✗	✗
FDFI	✓	✗	✓	✗	✗	✗	✗
CPMICF	✗	✗	✓	✓	✗	✗	✗
DFIPM	✓	✗	✓	✗	✗	✗	✗
DFMMIP	✗	✗	✗	✗	✓	✗	✗
GCFIM	✓	✓	✓	✗	✓	✗	✓
SDFIM	✓	✗	✗	✗	✗	✗	✗
PFP	✗	✗	✗	✗	✗	✗	✗
MHEI	✓	✗	✓	✗	✓	✗	✗

Table 4.2: Pre-investigation

The **pre-investigation** is a phase containing all steps that occur before the investigation itself. This phase has several purposes. First, the pre-investigation increases the efficiency of the investigation by training the investigation team, and providing adapted and up-to-date tools to them. Second, this phase allows to enhance the quality and the quantity of footprints by setting up devices to capture information upstream of the incident (security cameras, logs, etc.). Finally, this phase also aims to detect incidents and

alert adequate staff (appropriate number and with adequate tools to carry out the investigation). Table 4.2 evaluates the capacity of each approach to fulfil these objectives by stating the tasks that are included in each model.

Operations Readiness & Infrastructure Readiness: This step occurs before the detection of an incident and ensures that the investigative team is ready for any and all possibilities. Defined for the first time in (Carrier et al., 2003b), operations readiness is a preparation aiming to train (i.e. deepen, refresh or extend their knowledge and skills) personnel composing the investigative team (e.g. analysts, responders, etc.) and give them adequate and up to date equipment and tools. This preparation consists of several components, including staff training (awareness of anti-forensic techniques, training on the use of forensic tools, technology watch to ensure that investigators are qualified enough to work on the latest digital equipments, etc.), the acquisition and updating of efficient and approved tools and staff training on the laws in force in the geographical area concerned. As operations readiness, infrastructure readiness occurs before the detection of an incident. As (Carrier et al., 2003b) said, the infrastructure readiness to make sure that mechanisms are in place to ensure that enough information is collected upstream of the incident to allow the investigation to progress (Carrier et al., 2003b). This step is a preparation to maximise the number and the quality of evidence (Beebe and Clark, 2005). This phase may include the deployment of various hardware and software tools used to collect data about people, processes and their behaviours. Mechanisms that can be used include security cameras, log files on servers to trace network activities, door locks recording comings and goings, etc. This phase also includes the use of devices allowing the detection (or prevention) of abnormal behaviours (e.g. firewall, intrusion detection system, etc.).

Detection & Alert: This step marks the beginning of the work of investigators. The incident is identified using automated methods such as profile detection, anomaly detection or intrusion detection system or other methods such as the visual identification of a witness on the crime scene. After the detection, an alert is quickly sent to the appropriate organisation by using phone calls or automatic alert sent by software such as intrusion detection tools. The message sent by the witness or the system which detects the incident should be concise and accurate. Indeed, the alert must enable the responder to understand the situation, determine the nature of the incident and its gravity (Reith et al., 2002) (Shin, 2008). This understanding enables him to handle the case appropriately (send appropriate investigators regarding their skills and equip them with adequate tools, deploy the right number of investigators and the right amount of materials). We emphasise the importance of the alert that contributes, when properly formulated, to save valuable time by properly sizing the emergency team and the investigative team.

Confirmation & Authorisation: This step is intended to avoid false alarms. Using information contained in the alert message and information collected (questions asked to witnesses by phone, etc.), the incident is confirmed or not (Freiling and Schwittay, 2007). The authorisation step then aims to acquire authorisation from legal authorities or other entities (e.g. companies) to investigate the crime scene and seize materials required to carry out the investigation (Carrier et al., 2003b).

Preparation & Notification: During this step, the investigative team prepares forensic tools and software based on information obtained in the alert message (equipment to be used may vary depending on the type of the investigation). The objective of the notification is then to notify subjects of the investigation and third part that an investigation is starting. During police interventions with the aim of caught in the act, "surprise is needed to prevent destruction of evidence" (Ciardhuáin, 2004). For this type of interventions in particular, the step of notification is unsuitable.

4.2.2/ INVESTIGATION CORE

The **investigation core** is the main step of the investigation and starts when the investigators arrive on the crime scene. Among other steps, this part of the investigation is intended to protect the crime scene, search, collect and examine evidence and build the scenario of the incident. Table 4.3 evaluates the capacity of each approach to fulfil these objectives by stating the tasks that are included in each model.

Planning: After arriving on the crime scene, investigators choose a strategy allowing to efficiently carry out the investigation while "minimising the impact to the victims" (Reith et al., 2002). This step is intended to determine the objectives of the investigation ("what is known and what the investigators need to know") and ranked them in terms of importance (e.g. most volatile evidences need to be investigate first, most relevant evidence first, etc.) (Rogers et al., 2006). At the end of this step, the investigators know the objectives of the investigation, the systems that need to be investigated and in which order they have to be investigated. A strategy defines the nature and the sequencing of each task of the investigation in addition to how and by whom each task is carried out.

Live Response: Live response is a specific step of collection occurring before digital devices are turned off. The goal of this step is to collect volatile data (from machines which are still running) before they are erased by the shutdown of a machine (Freiling and Schwittay, 2007).

ID	Planning	Live response	Protection	Crime scene documentation	Search	Collection	Reduction & Organisation	Seizure	Storage	Evidence documentation	Examination	Event reconstruction	Timeline analysis	Tracking
CFIP	×	×	×	×	×	×	×	×	×	✓	✓	×	×	×
DFRWS	×	×	✓	×	×	✓	×	×	×	×	✓	✓	×	×
ADFM	✓	×	✓	×	×	✓	×	×	×	✓	✓	✓	×	×
IDIP	×	×	✓	✓	✓	×	×	×	×	×	✓	✓	×	×
EEDI	×	×	×	×	×	✓	×	×	×	×	✓	✓	✓	×
EDFIF	×	×	✓	✓	×	×	×	×	×	×	✓	✓	×	×
EIDIP	×	×	×	×	×	×	×	×	×	×	✓	✓	×	×
EMCI	✓	×	×	×	✓	✓	×	✓	✓	×	✓	✓	×	×
HOFDIP	✓	×	×	×	×	✓	×	×	×	×	✓	✓	×	×
CFFTPM	✓	×	×	×	✓	×	×	×	×	×	×	×	×	×
FDFI	✓	×	×	×	✓	✓	×	✓	✓	×	✓	✓	×	×
CPMICF	✓	✓	✓	×	×	✓	✓	×	×	×	×	✓	✓	×
DFIPM	×	×	✓	×	×	✓	×	✓	×	×	×	✓	✓	✓
DFMMIP	×	×	×	×	✓	×	×	✓	✓	×	×	✓	×	×
GCFIM	✓	×	✓	×	✓	✓	×	✓	✓	×	✓	✓	×	×
SDFIM	×	×	✓	✓	✓	✓	×	✓	×	×	✓	✓	×	×
PFP	×	×	✓	×	×	✓	×	×	×	×	✓	✓	×	×
MHEI	×	×	✓	×	✓	✓	×	✓	✓	×	✓	✓	×	×

Table 4.3: Investigation core

Protection: This step is one of the most important of an investigation as it ensures the integrity of the objects composing the crime scene (Palmer, 2001). Ensure the integrity of objects means secure the crime scene by preserving the current state of evidence (digital evidence and physical evidence) (Reith et al., 2002). To do so, the investigators stop processes and machines in addition to communication between devices and networks. In addition to preserve evidence, this step also aims to arrest suspects, to identify witnesses (Carrier et al., 2003b), to ensure the safety of people and to restrict the access of the crime scene (Agarwal et al., 2011). Finally, the protection phase also includes the generation of copies of the content of all digital devices used in the investigation to ensure the integrity of original data (Freiling and Schwittay, 2007).

Crime Scene Documentation: After ensuring the integrity of the crime scene, the investigators start to capture as much as possible information about the crime scene and the objects composing it (Carrier et al., 2003b). The documentation of the crime scene includes taking pictures, making digital records, etc.

Search & Collection: During the search step, the investigators search the crime scene to identify relevant objects which can be useful to reach the objectives of the investigation (Perumal, 2009). Objects identified during this step are potential source of digital or physical evidence. The investigators can also start to develop an initial theory about

the incident using objects identified (Carrier et al., 2003b). After identifying potential sources of evidences, the investigators start to collect those physical and digital objects. As defined in (Palmer, 2001), the goal of the collection is to gather information which may be used during the investigation using approved software and specialised algorithms enabling to recover hidden or erased data. In a digital context, data can be collected from storage media (e.g. hard disk, USB key, etc.) or by analysing network logs or capturing network traffic (Shin, 2008).

Reduction & Organisation: The collection step often results in the creation of a large dataset which is difficult to handle and analyse. To reduce the amount of data to process and facilitate its manipulation, the reduction and organisation step removes irrelevant data regarding the objectives of the investigation (Freiling and Schwittay, 2007).

Seizure, Transport, Storage & Evidence Documentation: Once all materials are collected from the crime scene, the investigators may transport some evidence to a location (e.g. a crime lab) providing suitable tools and person to perform an in-depth analysis of them (Ciardhuáin, 2004)(Shin, 2008). To ensure the integrity of evidence during the transport, measures should be taken such as proper packaging of evidence (Vlachopoulos et al., 2013). In case the in-depth analysis cannot be carried out immediately, the evidences are stored in a proper and locked room to ensure its integrity (Ciardhuáin, 2004). At the same time, each evidence is documented to describe where it comes from and how it was collected (Pollitt, 1995).

Examination: The examination step aims to identify what each evidence is about. In case of a digital document, for example, it means read and understand it and determine if the content of the evidence is relevant for the investigation regarding its objectives (Pollitt, 1995). In (Agarwal et al., 2011), the author said that "this phase aims at making the evidence visible, while explaining its originality and significance" and defined the examination as a study of evidence by forensic specialists to extract relevant information to solve the case. To carry out the examination of a digital evidence, techniques such as pattern matching or filtering can be used (Palmer, 2001).

Event Reconstruction & Timeline Analysis: The event reconstruction phase is one of the most time consuming step of the investigation. It consists in the analysis of information extracted during previous steps to identify events which happened during the incident (Palmer, 2001). At the end of the event reconstruction, a timeline containing all events which occurred during the incident is generated. This timeline is then interpreted and analysed to extract the scenario of the incident, understand the circumstances of it (identify the perpetrator and his motivation, etc.) (Freiling and Schwittay, 2007) and produce

evidence to refute or confirm allegations about the incident (Beebe and Clark, 2005).

Tracking: Using information collected about the perpetrators of the incident, the investigators track and arrest suspects (Shin, 2008).

4.2.3/ POST-INVESTIGATION

The **post-investigation** closes the investigation. This phase marks the end of the investigation by presenting and defending the results of it in a court and reviewing the investigation to improve processes for further investigations. Table 4.4 evaluates the capacity of each approach to fulfil these objectives by stating the tasks that are included in each model.

Report: Prior to being presented in court, the results of the investigation need to be formatted and organised to be easily viewed and understandable. (Freiling and Schwittay, 2007) recommends to write a report which describes the incident and which contains the conclusions of the investigation and documentation about all evidence found during it. In (Shin, 2008), the authors also add that this report may contain information about suspects and methods use to commit the crime. Finally, this report should contain information to explain the reasoning process used to lead to each conclusion.

Presentation & Defence: After formatting and documenting the results of the investigation to make them admissible, they are presented to justice (Palmer, 2001). The format of the communication depends on the type of audience (justice, technical personnel of a company, legal personnel of a company, etc.) (Beebe and Clark, 2005). Once the results are in possession of justice, then began a discussion among participants. Investigators ensure the defence of their theory using evidence to support the validity of it (Ciardhuáin, 2004). During this step, explanations of the methods used to reach each conclusion are given to improve the credibility of the results (Pilli et al., 2010).

Resolution & Restitution of evidence: Using results presented, the verdict is given by the court (Palmer, 2001). The resolution step is intended to solve the case, to take sanctions against people responsible of the incident and to take measures to avoid the same kind of incident to occur again (Freiling and Schwittay, 2007). At the end, the evidences seized during the investigation are returned to its owner (when it is possible) (Reith et al., 2002).

ID	Report	Presentation	Defence	Resolution	Restitution of evidence	Review
CFIP	x	✓	x	x	x	x
DFRWS	x	✓	x	✓	x	x
ADFM	x	✓	x	x	✓	x
IDIP	x	✓	x	x	x	✓
EEDI	x	x	x	x	x	x
EDFIF	x	✓	x	x	x	✓
EIDIP	x	✓	x	x	x	✓
EMCI	x	✓	✓	x	x	✓
HOFDIP	x	✓	x	x	✓	✓
CFFTP	x	x	x	x	x	x
FDFI	x	✓	✓	x	x	x
CPMICF	✓	x	x	✓	x	x
DFIPM	✓	✓	x	x	x	x
DFMMIP	x	✓	✓	x	x	✓
GCFIM	✓	✓	✓	✓	✓	✓
SDFIM	x	✓	x	x	x	✓
PFP	x	✓	x	x	x	x
MHEI	x	x	x	x	x	✓

Table 4.4: Post-investigation

Review: The review is the last step of the investigation. As (Carrier et al., 2003b) explains, this step aims to review the investigation to identify possible improvements for future investigation. The results of this step may be new procedures, training or tools for future investigations. This step also allows to disseminate information about the investigation to enable other investigators to gain experience and new knowledge from it (Ciardhuáin, 2004).

4.2.4/ OTHER WORKS

In the previous section, we have presented a synthesis of the most common and significant steps that can be found in existing models. This section focuses on three models which introduce original elements for the investigative process in the margins of more conventional models.

In the proposal of (Carrier et al., 2003b), the author introduces the idea that a computer can be seen as a digital crime scene. The authors explain that "instead of treating the computer as a substance that needed to be identified, it is treated as a secondary crime scene". Therefore, the investigation process model used for physical investigation can be used in a digital context. The proposed model is called Integrated Digital Investigation Process (IDIP). It is made of five main phases (commonly used in the other process models) and is illustrated in Figure 4.1.

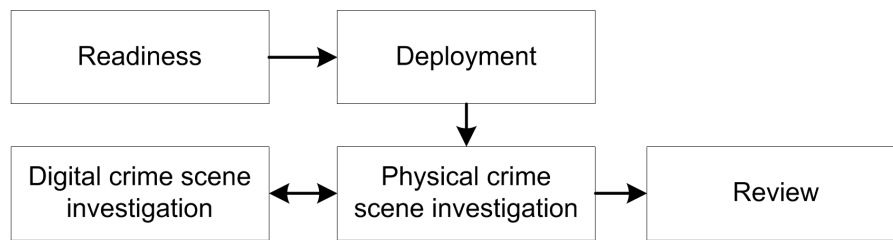


Figure 4.1: Integrated Digital Investigation Process (IDIP)

The physical crime scene investigation phase is intended to collect and analyse footprints left in the physical crime scene to reconstruct the scenario of the incident. The investigation of a physical crime scene is made of seven steps from the preservation of the crime scene, to the presentation of the final theory (illustrated in Figure 4.2).

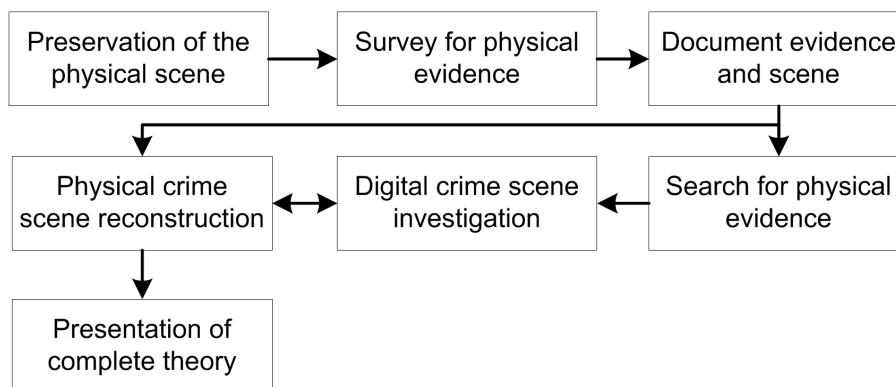


Figure 4.2: Physical crime scene investigation in IDIP

The digital crime scene investigation aims to collect and analyse footprints left on the digital crime scene to reconstruct the scenario of the incident. During the physical investigation, when a digital device is found, a digital investigation starts using the device as a digital crime scene. The process of digital investigation is illustrated in Figure 4.3.

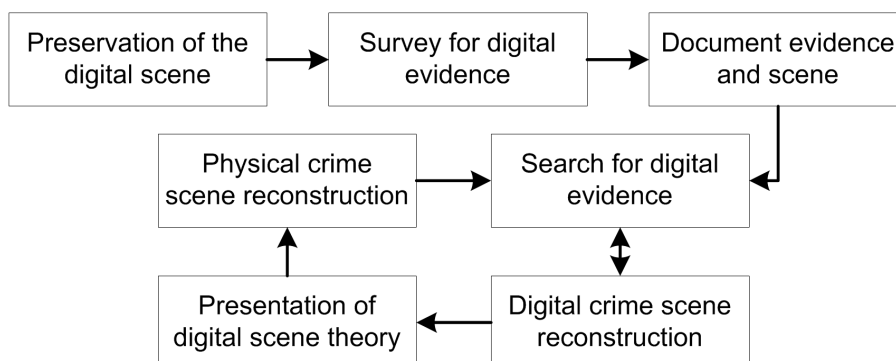


Figure 4.3: Digital crime scene investigation in IDIP

The results of each digital investigation are injected and used in the physical investigation to build a global theory about the incident. The originality of this model lies in the distinc-

tion between physical crime scene investigation and digital crime scene investigation and the integration of these two notions in a single process.

In (Carrier and Spafford, 2004a) and (Carrier and Spafford, 2004b), the authors introduce a high-level process model as well as more detailed models for parts of the process as the event reconstruction. This model, called Event-based Digital Forensic Investigation Framework (EDFIF), derives from earlier work of Carrier on the model IDIP (Carrier et al., 2003a) and is illustrated in Figure 4.4. An investigation in a digital crime scene is made of three steps which are: "System Preservation and Documentation Phase", "Evidence Searching and Documentation Phase" (illustrated in Figure 4.5) and "Event Reconstruction and Documentation Phase". The target definition defines what investigators are looking for (e.g. a particular type of files) based on their experience and evidence already found. The next step consists in the extraction of objects which satisfied the target (e.g. if the investigators are looking for image, this step extracts JPEG and PNG files for example). Following data extraction, data extracted is compared with the target to select objects which can be used as evidence. At the end of an iteration of the search process, the definition of the target may be updated using knowledge acquired from new evidence.

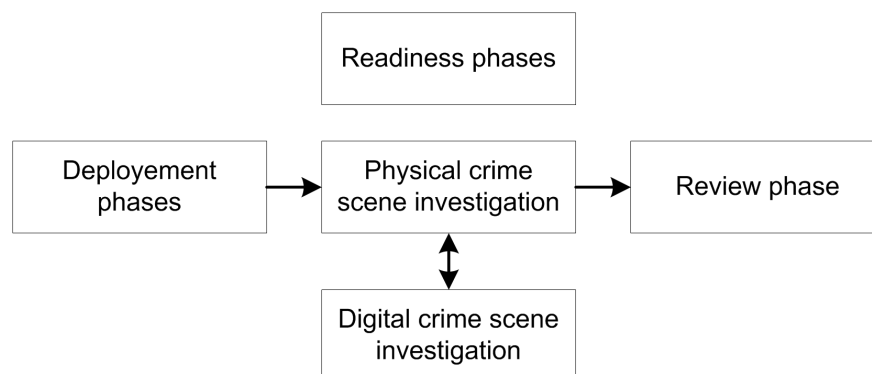


Figure 4.4: Event-based Digital Forensic Investigation Framework (EDFIF)

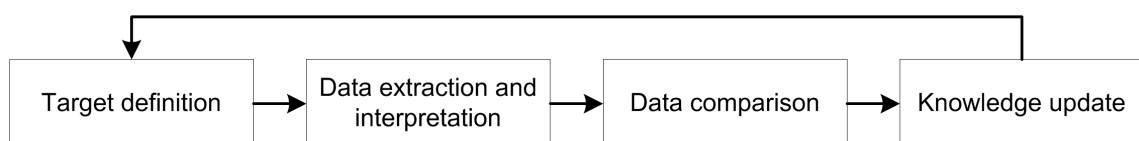


Figure 4.5: Evidence searching in EDFIF

The next phase of an investigation of a digital crime scene is the event reconstruction, illustrated in Figure 4.6. It consists in study evidence to determine events which happened during the incident.

To carry out the reconstruction, investigators first examine each evidence to determine what was its role in the incident. The event reconstruction then enables to determine events in which evidence are involved (an evidence is at least the effect of one event). During the event sequencing, a chain of events is built. To conclude, the validity of the

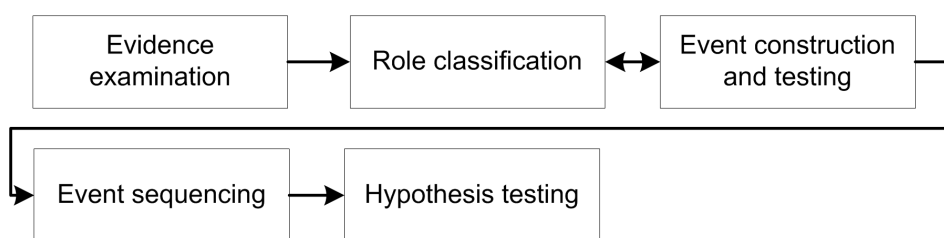


Figure 4.6: Event reconstruction in EDFIF

scenario is tested during the hypothesis testing phase.

In (Stephenson, 2003), a process model based on (Palmer, 2001) and detailing precisely the event reconstruction phase is proposed (this model is illustrated in Figure 4.7). This model, called End to End Digital Investigation (EEDI), is composed of nine main steps. This process model is not complete and should be used in conjunction with the DRFWS model.

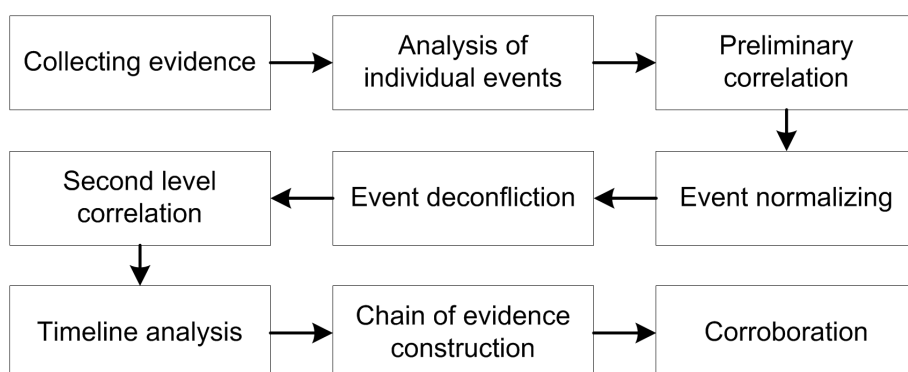


Figure 4.7: End to End Digital Investigation (EEDI)

The first step of the EEDI process is intended to collect evidence and to deduce a collection of events which are the atomic elements composing an incident. In their approach, the authors are not focusing only on events related to the incident. Indeed, some of the events may seem insignificant if they are considered in a separate way. However, if an event is studied in a context consisting of other events, it may become important for the investigation. After identifying all events from the evidences collected, the investigators examine each event to determine its value regarding the objectives of the investigation. Following the analysis of individual events, they correlate events into a chain to understand what happened (in broad terms) and eliminate duplicate events (an event can be reported in several sources) and conflicting events (i.e. find events that are reported multiple times in the same source and consider it as an individual event). After normalisation and deconfliction, events are used to build a timeline. This step in addition to normalisation, deconfliction and correlation constitutes an iterative process. From the timeline, a coherent chain of events is then extracted to form the scenario of the incident. To conclude, the corroboration phase enables to corroborate the events composing the timeline with

new evidence that are not yet take into account in the investigation.

4.3/ DISCUSSION

As we argued previously, the introduction of our approach for event reconstruction requires to integrate it into the investigation process. Indeed, our tools must fit properly in the process that investigators use and must be interfaced satisfactorily with the steps upstream and downstream of the process. Furthermore, as we state in Section 4.1, the definition of a process model allows to partially answer the problems identified in the previous chapter 1) by explaining the steps of the investigation process and thus contribute to the reproducibility and the credibility of this process and 2) by providing a framework for the development of automated software tools able to handle large volumes of data as well as guidelines for human investigators. The aim of this discussion is therefore to identify strengths and limitations that need to be overcome to reach these goals in order to propose a new process model addressing their shortcomings.

First, an accurate and highly detailed definition of the investigation process is needed, as explained in Section 4.1. Concerning this point, the situation has improved significantly in recent years. In 2002, (Reith et al., 2002) state that "there does not exist a standard or consistent digital forensics methodology, but rather a set of procedures and tools built from the experiences of law enforcement, system administrators, and hackers". Since this observation, substantial efforts have been made in the development of process model allowing to give credibility to investigations by describing the processes used in a precise and exhaustive way. The models presented in this chapter cover and explain all aspects of a digital investigation from the detection of the incident to the resolution of the incident. They describe the most important steps and milestones of a digital investigation and therefore give guiding principle to investigators. Despite this, they are still not accurate enough because steps used in models are too abstract to clearly explain what is done in each of them. This state of the art shows that the majority of the process models are high level models and designed to guide investigators. In our work, we seek to develop software tools, therefore a more complete and accurate model is needed. Especially, "existing models do not explicitly identify information flows in investigations" (Ciardhuáin, 2004). This is especially true for the event reconstruction step. In existing models, this latter is explained briefly in natural language while a very detailed and precise definition of the process is needed. Thus, further efforts must be done to even more clarify the process, especially if we want to use it as a framework to guide the development of digital forensics software tools. Second, due to the large quantity of models proposed so far, it may be difficult for developers to select the appropriate model to answer their needs. In addition, our state of the art show that there are many redundancies between models. A large majority of models used as a backbone the scheme proposed in (Palmer,

2001): "Detection of an incident", "Protection", "Collection", "Examination", "Analysis", "Presentation", "Resolution" in addition to one or more steps to enrich this model. Several other models introduce different approaches such as (Carrier et al., 2003b), (Stephenson, 2003) and (Carrier and Spafford, 2004a), (Carrier and Spafford, 2004b). We argue that it is necessary to propose a model synthesising the qualities of existing models to obtain a unique investigation process widely accepted within the digital forensics community. To answer these two limits, this thesis introduces a new process model in Chapter 6.

4.4/ CONCLUSION OF THE STATE OF THE ART

In the three chapters constituting the state of the art, a detailed study of the problem of reconstruction of events has been proposed. Chapter 2 has first placed this problem in context and highlighted the importance of the reconstruction for a digital investigation. During the reconstruction, investigators face three major issues that are the amount of data to process, the heterogeneity of these data and the legal requirements that must be respected to ensure the admissibility of the results at trial.

In Chapter 3, ten event reconstruction solutions were studied in the light of seven characteristics deemed essential by the authors to answer the issues inherent to the reconstruction problem. This study has shown that it remains significant problems to address. First, the state of the art has highlighted that the existing analytical tools have low adaptive capacity. In particular, rules-based approaches can not adapt to unknown cases and the definition of the set of rules is a tedious task. Second, the data structures used to store the knowledge collected from the crime scene are not accurate and complete enough to represent faithfully digital events. Therefore, the analysis capabilities of approaches based on these data structures are thereby reduced. Finally, the literature contains many proposals to ensure the traceability of the information and the credibility of the results. However, these two features are not present both in a unique approach. Moreover, we have identified two approaches based on widely recognised theories ensuring the credibility of results. However, these approaches have significant limitations on other aspects such as their ability to handle large volumes of data.

To overcome the limitations of the existing, an innovative approach, based on knowledge engineering techniques, is introduced in Chapter 5 to overcome the limitations of existing approaches. In Chapter 6, a novel digital investigation process model is presented to integrate this approach in the broader process that is the digital investigation.

SEMANTIC ANALYSIS OF DIGITAL FORENSIC CASES

APPROACH

The reconstruction of events is a problem addressed by several approaches as shown in the state of the art of Chapter 3. Despite the relevance of the existing approaches, obstacles remain and need to be solved in order to construct an approach meeting the expectations of both investigators and judges. Regarding the challenge related to the volume of data to handle and its heterogeneity, the state of the art shows that it exists efficient tools to extract information from large and heterogeneous sources of information (e.g. Plaso). However, the storage of all this knowledge, its analysis and the visualisation of these large data volumes remain problems that are partially solved by the existing tools. Moreover, the issues inherent to the legal requirements are not sufficiently treated in the literature and few solutions are proposed.

To solve the remaining problems, this chapter introduces an approach, named SADFC, taking advantage of techniques and methods from the field of knowledge engineering. The central idea of SADFC is to manage the semantics of the data to help the investigators to face the cognitive overload during the analysis of the data while ensuring the quality of the results, from a legal point of view. The development of this approach is organised in two research axes. To address the aspects related to cognitive overload, a first research axis aims at providing a data structure for the representation of the semantics of the data in order to build more advanced analysis tools able to assist the investigators. To meet the legal requirements, a second research axis is to ensure the credibility and the traceability of the results through the development of a formal theory for the reconstruction of events.

This chapter is organised as follows: Section 5.1 presents the SADFC approach and gives an overview of its structure. Section 5.2 presents the component regarding the first axis. This section especially aims to demonstrate the potential of the ontology. Section 5.3 finally introduces the components inherent to the second axis.

5.1/ SEMANTIC ANALYSIS OF DIGITAL FORENSICS CASES

To address the issues identified in the state of the art, we introduce in this thesis the SADFC approach. SADFC is a semantic approach for representing the knowledge related to an incident and the associated investigation. The final goal of this approach is to produce, automatically and in a timely manner, results that are admissible in a court. Figure 5.1 shows the relations between the components of SADFC and the requirements identified in Chapter 3. Each of these components allows to solve one or more of these requirements. Some of these requirements are research issues for which we propose innovative solutions. These solutions are described in the following sections. The other requirements are technical locks, i.e., issues solved in the existing works that we have to implement and to integrate in our approach while taking into account its specificities. For example, the extraction of information from heterogeneous sources is a problem solved in most of the existing approaches. However, this functionality has to be integrated in our own approach as it is an essential feature for the proper functioning of it. The integration have, however, to take into account the specificities of our approach, i.e. in this case, the extraction of information must format the data so that it is usable by the other elements of our approach.

The SADFC approach consists of several components briefly introduced in this chapter and described in detail in the following chapters of this manuscript. It is a synergy, illustrated in Figure 5.1, of elements which, once assembled, constitute a coherent package describing methods, processes and technological solutions needed for event reconstruction. The first component is a formalisation of the problem of event reconstruction (see component (1) in Figure 5.1). This formalisation describes the entities composing a crime scene, including footprints, events, resources and protagonists. Formal operators based on this formalisation (see component (2) in Figure 5.1) are defined to complete the event reconstruction, from the extraction of footprints to the analysis of information. These two components are introduced briefly in Section 5.3 and presented in detail in Chapter 6. The second component is a definition of the investigation process model in which we integrate our event reconstruction tools (see component (3) in Figure 5.1). The aim of this model is to define the various phases of the event reconstruction process: their types, the order between them and the data flow through the whole process. The definition of this process model leverages formal operators defined in the previous component. This component is also introduced in Section 5.3 and presented in detail in Chapter 6. The third and last component is an architecture implementing the theoretical components of the approach (see component (4) in Figure 5.1). This architecture is composed of several modules, each implementing functions required for the reconstruction of events. This includes footprint extraction, knowledge management, knowledge analysis and visualisation. The proposed architecture is centred on an ontology (see component (5) in Figure 5.1) enabling to model in a structured and precise way a digital incident and the

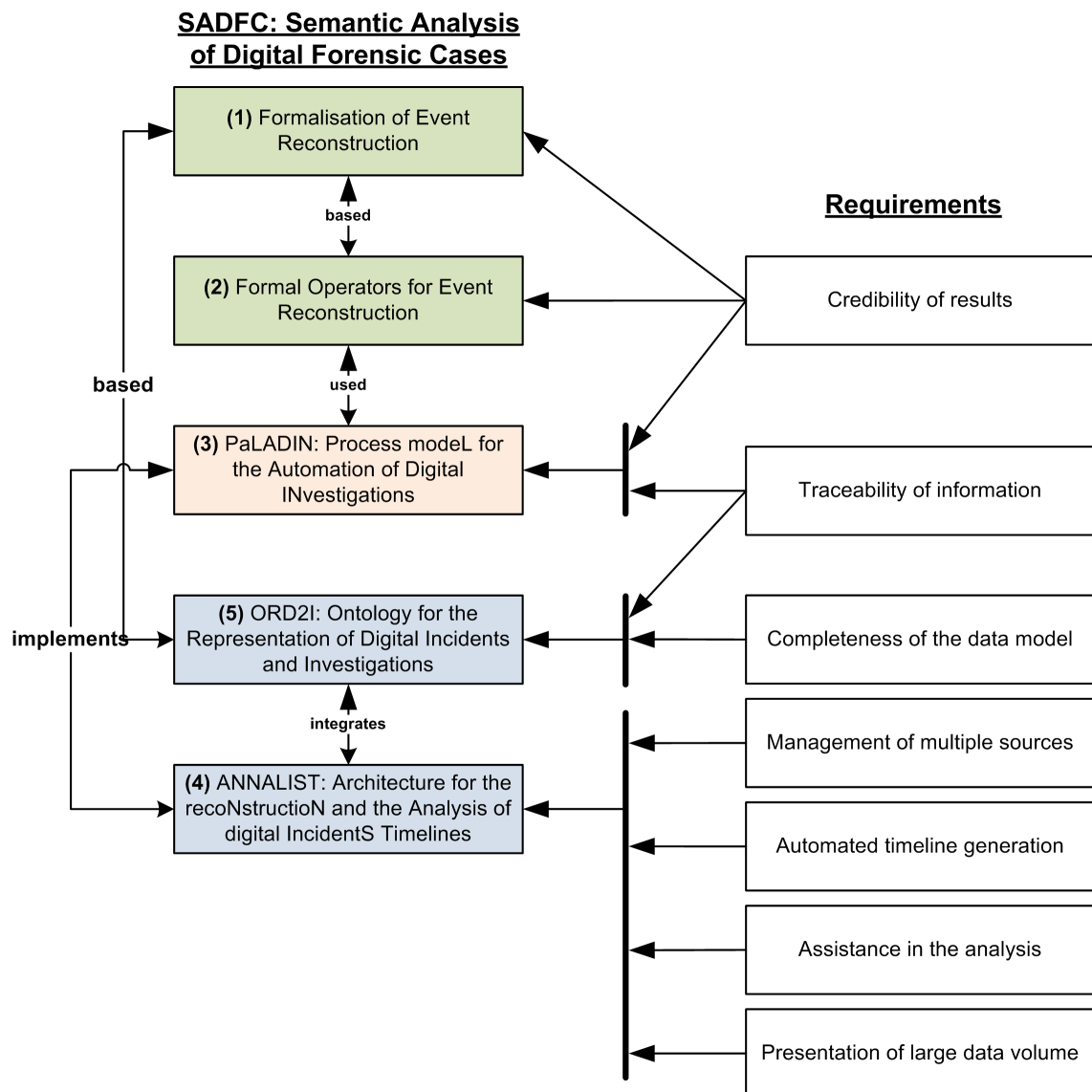


Figure 5.1: Interactions between the components of the SADFC approach

associated investigation. These two components are outlined in Section 5.2. The ontology is subsequently presented in detail in Chapter 7. The architecture, for its part, is presented in Chapter 8.

5.2/ CONTRIBUTION OF SEMANTICS FOR ANALYSIS OF INCIDENTS

The state of the art has shown that the interpretation of data in a crime scene is difficult for investigators, because of the large volumes involved and their heterogeneity. The analysis of large volumes of data can not be conducted only by investigators as they are limited by their processing capacity. In consequence, they are facing cognitive overload. It is therefore essential to provide investigators with tools for viewing and manipulating these

large volumes of data and assist them in the analysis of the latter. The objective is to guide the investigator by highlighting potentially relevant information in the gangue of data to achieve the objectives of the investigation, but without fully replacing him. Indeed, the expertise and experience of the investigators are required to carry out the most complex analysis tasks.

To develop such tools, it is necessary that the analysis algorithms have the ability to understand the data on which they work. During the event reconstruction process, the investigators try to reconstruct the circumstances of the incident from the digital footprints left in a crime scene. Thereafter, the analysis of the timeline of the accident allows to determine the responsibilities of the protagonists and the *modus operandi* of the criminal. These cognitive processes involved can be compared to the semiotic triangle proposed in (Ogden et al., 1923) and represented in Figure 5.2. The semiotic triangle described the interactions between the sign, the object and the concept.

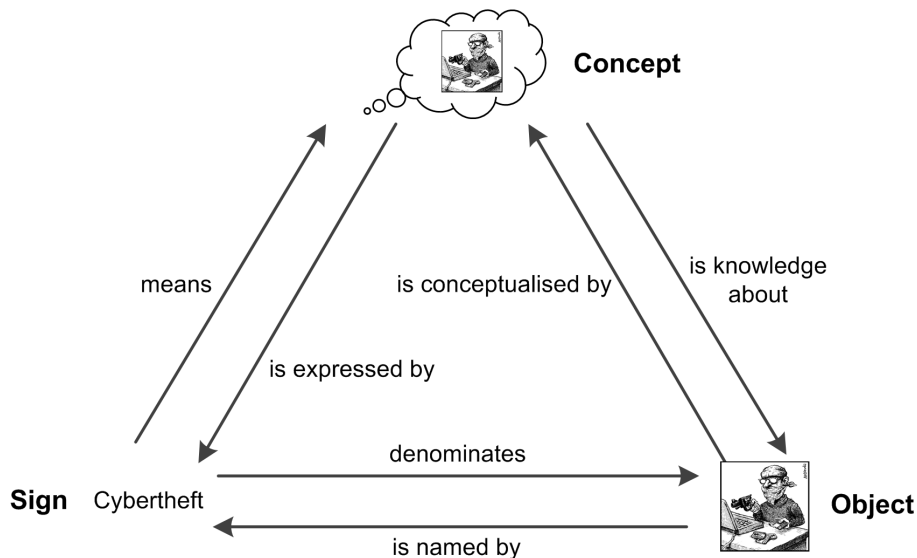


Figure 5.2: Semiotic triangle

- The sign is used to describe an entity of the world. A sign is part of a larger set called syntax. In the context of event reconstruction, the sign is comparable to the footprints found in the crime scene. The footprints collected at the beginning of the investigation have no meaning for investigators and their goal is to interpret them and give them a meaning to understand what happened during the incident.
- The object is what is referred by a sign. The footprints left in a crime scene describe the entities related to the events occurring in the scene. In the context of event reconstruction, the object is equivalent to these entities.
- The concept is part of a larger set called semantics. Semantics, as opposed to syntax, define the mental representation of entities corresponding to the signs used in texts or images. A concept is the meaning of an entity and is relative to the point of view of the person formulating this meaning. Indeed, the same entity is

not perceived in the same way by two distinct people. In the context of event reconstruction, the concept is the interpretation made by the investigators using the information found in the crime scene. This interpretation is a subjective vision of the incident, more or less faithful to reality. Indeed, the footprints left in the crime scene may only allow a partial reconstruction of the incident or a distorted or biased vision of it.

During the reconstruction of events, the investigators try to determine the elements (actions of the protagonists, items used, etc.) composing the incident (i.e., the objects) using the information contained in the different available sources (i.e., the signs) such as log files, web histories, etc. The semantisation of the footprints enables them to build an interpretation of the incident (i.e., the concept). After extracting the knowledge about the incident, the investigators finally issue conclusions about it.

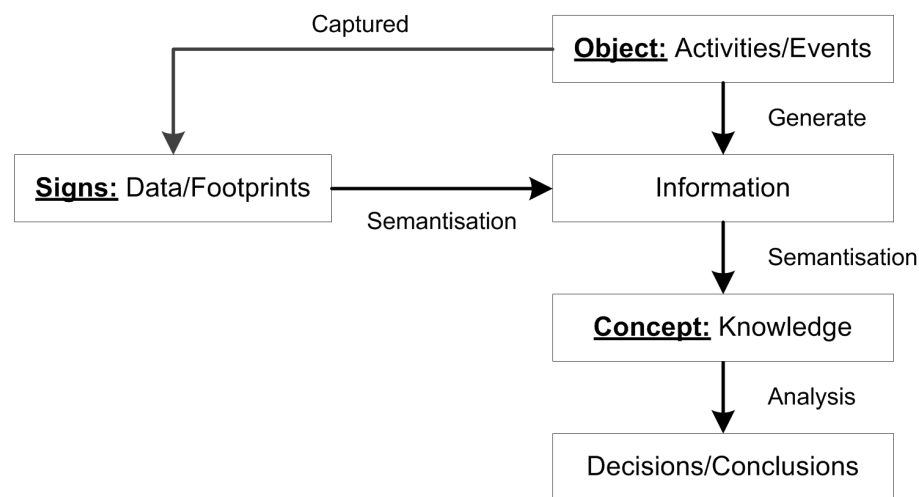


Figure 5.3: Event reconstruction process

This process is illustrated in Figure 5.3 (this figure is derived from (Liew, 2007)). During it, the investigators give to each footprint a meaning that allows them to better understand the incident and to have a more complete vision of it. This process is comparable to the reading of a text. When a person reads a text, he uses a semantisation process which enables him to associate an interpretation to each sign identified. This operation uses a number of underlying processes which are trivial for a human but very complex for machines.

The goal of the first research axis of our project is to develop tools to perform automatically the semantisation process and the analysis of the footprints, i.e., the process starting from the signs and ending with the conclusions drawn by the investigators in the light of the knowledge collected during the investigation. The first challenge is to build a process able to extract the meaning of footprints. To do so, it is necessary to propose a semantisation step to reassign the original meaning of the data. This process moves from signs with no meaning to semantic concepts connected to each other by semantic

properties representing the events that occurred in the past. The construction of such a process involves two elements. The first element is a knowledge model to store knowledge about the entities composing the crime scene. The second element is a tool taking as input the footprints and outputting an interpretation of the incident which will populate the knowledge model. This knowledge is then used by intelligent and automated analysis processes able to draw conclusions by taking advantage of semantic information. Finally, because of the large volumes of data involved, it is also essential to provide tools to visualise and manipulate data efficiently.

5.2.1/ KNOWLEDGE MODEL FOR EVENT RECONSTRUCTION

The choice of the knowledge representation model is crucial to offer advanced analysis capabilities. Currently, a large number of forensic tools (Mactime (Farmer and Venema, 2004) for example) works on unstructured data as plain text which do not allow to build advanced analysis process. This is due to the lack of semantic information that do not enable machines to understand the meaning of data. A promising perspective is to use a precise and reliable representation, enabling to structure data on the one hand, and to standardise its representation on the other hand (in order to deal with heterogeneity issues). Introducing a structured and formal knowledge representation pursues two goals: build automated processes more easily by making information understandable by machines and make data easy to use and understandable by human investigators (graph visualisation, query tools, etc.).

For these purposes, a knowledge base is used as the central element of our approach. It aims to store complex information in a structured way and to reason on this information (i.e. deduction of new knowledge, identification of inconsistencies, etc.). A knowledge base is therefore a relevant structure to store all the information extracted from the crime scene and to reason on it. It is made of two elements: a schema structuring the knowledge and individuals instantiating this schema (Hepp, 2008). The schema defines the structure (i.e. the terminology) used to model the knowledge. In our work, the knowledge base is structured using an ontology and instantiated using the knowledge collected from the crime scene. This ontology, named ORD2I, is implemented using OWL 2¹, a language based on description logic and allows to model accurately a digital incident. Gruber defines an ontology as "an explicit specification of a conceptualisation" (Gruber et al., 1993). The explicit and formal nature of ontology facilitates the design and the use of interpretation and analysis tools. These two characteristics allow the construction of vocabularies comprehensible by the machine for describing knowledge. Thus, the analysis algorithms using the vocabulary acquire an understanding of the data to be processed. In particular, ontology languages such as RDF or OWL enable to model the meaning of information (RDF and OWL are interested in the meaning of data when other formats such

¹<http://www.w3.org/TR/owl2-overview/>

as XML is interested in structuring data). Thus, using ontology languages, machines are able to understand the meaning of data without prior knowledge about the domain.

The ontology also helps to meet the need of completeness. This ontology is the receptacle of the knowledge collected from the crime scene and provides to analysis and visualisation tools the necessary data to accomplish their goal. Indeed, the knowledge model must be complete and accurate enough to represent faithfully and completely (every information potentially valuable to reach the objectives of the investigation) the knowledge extracted by extraction tools from footprints. The completeness of the ontology also allows analysis tools to work on an accurate view of the incident and therefore produce more informed conclusions (the greater the number of input facts is, the greater the number and the quality of the deductions are). An ontology is a model enabling to represent knowledge of a given domain by structuring this knowledge using concepts, properties and logical constraints on them. It provides rich semantics to represent the knowledge (richer than databases due to its sophisticated semantic concepts (Martinez-Cruz et al., 2012)) Thus, ontologies are able to represent accurately the knowledge generated during an investigation (knowledge about footprints, events, protagonists etc.). Unlike more rudimentary data formats, ontology can represent relations between concepts in addition to the underlying logic of data.

Another advantage of ontology is to enable the development of generic analysis tools. These tools must be sufficiently generic to be adaptable to all types of situations, even the unknown cases. For this, we implement in our ontology two layers of knowledge. The first layer of knowledge contains common information on the entities composing the incident. This layer contains in particular temporal information and location and also describes the resources used by the events and protagonists involved in them. The second layer contains specialised and technical knowledge about the incident and especially the objects involved in it. For example, this layer models the metadata associated to a PDF file, the hash of a file or the sender or the receiver of an email. This separation between common knowledge and specialised knowledge allows to offer hybrid analysis approaches operating from assumptions on knowledge from both layers. Thus, even in the absence of technical information about the objects, our analysis tools can still operate using hypothesis on common knowledge (assumptions on time, location, etc.). Furthermore, the integration of a specialised knowledge layer enables further and advanced analysis by taking advantage of this knowledge when the information is available in the sources of information.

Finally, as stated in (Gruber et al., 1993), ontologies can be used to build a common view of a domain which means that this kind of data structure is particularly relevant to build a consensus on the representation of events among digital forensic investigators and software developers. In addition, ontologies have already proved its relevance in computer forensics (Schatz et al., 2004a,b). The use of ontology is also motivated by its successful use in other fields such as biology (Schulze-Kremer, 1998) and building life cycle man-

agement (Vanlande et al., 2008). Therefore, ontology is the data structure best suited to meet the requirements mentioned above because of its inherent characteristics. In the rest of the manuscript, it is assumed that the reader is familiar with the field of knowledge engineering and the semantic web technologies. If necessary, the reader can refer to Appendix A. This appendix explains the concepts of ontology and defines the related keywords used in the manuscript. It also presents the technologies used by our approach, including ontology language such as RDF, RDFS and OWL 2, and the query language SPARQL Protocol and Resource Description Framework Query Language (SPARQL).

5.2.2/ ARCHITECTURE

To carry out the reconstruction of events, this ontology must be associated with upstream and downstream tools to achieve the process starting from the crime scene and leading to the conclusions presented at trial. The first part of this process starts from the footprints collected to the storage of their meaning in the knowledge base. The second part of this process takes advantage of this knowledge about the crime scene to produce conclusions about the incident. In our approach, an architecture called Architecture for the recoNstruction and the Analysis of digital IncidentS Timelines (ANNALIST) is proposed to achieve this two-stage process. ANNALIST takes advantage of the technologies of the semantic web stack. These technologies offer, in particular, efficient means to instantiate the ontology, store the knowledge base, manipulate and query the knowledge and reason about it. This architecture, presented in Chapter 8, is made of four layers. Each of these layers provides functions carrying out a part of the reconstruction process.

The **extraction layer** provides tools to collect information from heterogeneous sources. In the state of the art of Chapter 3, we have seen that the extraction from large and heterogeneous sources is a problem solved by tools such as Plaso. Therefore, ANNALIST takes advantage of the Plaso toolbox to carry out the extraction process. The tools proposed by Plaso enable to extract information from a large panel of sources that can be found in disk images. This tool is very efficient as it can process a disk image of several gigabytes in minutes. To interface the output of Plaso with the following layer (**knowledge layer**) of ANNALIST, a set of bridges has been developed. The knowledge layer is intended to populate the ontology using the information collected by the extraction layer. The **reasoning layer** provides tools to assist the investigators during the analysis of timelines. The state of the art has shown that the existing analysis tools are limited by the chosen data structure or by the use of rule-based techniques. The use of such techniques does not allow to produce tools that can adapt to unknown situations involving new knowledge and the cost of the update is important. To address this issue, we introduce analysis tools based on two types of hypothesis: assumptions using technical knowledge (and thus, taking advantage of the completeness of the ORD2I ontology) and assumptions based on common knowledge such as time, location and unifying concepts

like the notion of resources or subjects. To enable the development of these tools, the ability of the ontology to structure knowledge in addition to the provision of two knowledge layers are two important assets. These analysis tools are formally described in Chapter 6 and their implementation is presented in Chapter 8. Finally, the **interface layer** provides tools for the manipulation and the visualisation of the knowledge contained in the knowledge base. The use of an ontology as backbone make very convenient the manipulation of data thanks to the use of tools such as SPARQL², a language designed to query graph knowledge. Due to the structure in the form of triples <subject, predicate, object>, it is also possible to visually represent ontologies in the form of graphs. This kind of graphical representation is very intuitive and clearer than a textual representation of data.

5.3/ CREDIBILITY AND TRACEABILITY

The credibility and the traceability are two key features to ensure the admissibility of evidence at trial. As shown in Chapter 3, few approaches offer satisfactory solutions to get these two characteristics. The approach proposed in (Gladyshev and Patel, 2004) is the only approach proposing a solution to guarantee the credibility of the results (this approach is based on finite state machine, a widely accepted theory). However, this approach is limited regarding other aspects, especially when comes the need to process large volumes of data. Regarding reproducibility, the most satisfactory solutions intended to memorise for each reasoning operation, the data used as input. This type of solution is used in particular in (Hargreaves and Patterson, 2012) (timeline summarisation) in which each high-level event is associated with the set of low-level events that allowed the identification of it. It is therefore possible to trace back the path from the results to the original information.

In Chapter 4, we argue that the level of credibility of an evidence is directly related to the accuracy and the verifiability of the methods used to produce it. In our approach, the credibility of the evidence is ensured by several elements. The main idea is to introduce a theoretical and formal foundations for the tools composing our approach. The first element ensuring the credibility is a formalisation of the problem of event reconstruction. This formalisation defines the concepts composing a crime scene and the relations between them. The introduction of such a formalisation disambiguates the notions handled by our approach and clarifies the operation of it. Moreover, it is also a starting point to build the ORD2I ontology. As stated in Section 5.2, an ontology is made of concepts and properties. The study of the data that can be collected in a crime scene leads to a formalisation of the concepts composing a crime scene and the properties linking these concepts. This formalisation contains the elements to be included in the ontology and therefore constitutes the foundation of the latter. The second element of this theoretical

²<http://www.w3.org/TR/rdf-sparql-query/>

base is a set of operators, formally defined, to carry out each part of the reconstruction of events (i.e., extraction/instantiation operators and analysis operators). The definition of these operators allows to clearly define the operations performed by them (input, output and the transformation made by each operator) on the entities of the crime scene and, thus, adds credibility to the results produced by them.

To ensure the reproducibility of an evidence, it is necessary to be able to explain how each evidence is produced. The SADFC approach ensures the traceability of the information using two components. We argue that the reproducibility of the results requires both a clear definition of the investigation process and the introduction of mechanisms to store information about the tasks performed to achieve results. The first element can be considered as a meta investigation process when the second element can be seen as an instantiation of this meta process. The first element takes the form of a formal process model, named Process model for the Automation of Digital INvestigations (PaLADIN), describing precisely each step composing the investigation and, thus, allowing the court members to understand the course of the investigation. The second element is an additional layer of knowledge integrated in the ORD2I ontology. This layer is dedicated to the representation of knowledge about the investigation itself. Each task composing it is modelled in this layer, including the information used as input and the result produced by the task. This layer gives the ability to explain to the users every reasoning used to produce new knowledge. It is thus possible to trace the path used to produce each result of the investigation.

5.4/ CONCLUSION

The approach proposed in this chapter takes advantage of techniques from the field of knowledge engineering to give answers to the needs identified in the state of the art. This approach, called SADFC, is composed of three main elements: a formalisation of the problem of event reconstruction and formal operators to carry out this task, an investigation process model (PaLADIN) and an architecture (ANNALIST) centred on an ontology (ORD2I). The formal part of the approach is intended to give credibility to the results by explaining the operation of reconstruction operators using mathematical definitions. The formalisation of the event reconstruction is also the foundation of the ANNALIST architecture and, more particularly, the foundation of the ORD2I ontology as it defines the concepts and the properties composing it.

The second part of SADFC is a comprehensive and accurate definition of the investigation process. This model aims to ensure the reproducibility and the credibility of the results by making explicit the steps composing the investigation. The second goal of this model is to ensure a proper integration of our tools in the larger picture which is the digital investigation.

The final part is an architecture centred on an ontology. This software architecture is the implementation of the theoretical components of the approach and allows to carry out the reconstruction on real cases. ANNALIST is made of a large set of tools allowing to extract information from disk images, to instantiate the ontology, to make deductions about the incident and help the investigators during the analysis of it and to visualise the knowledge contained in the base.

FORMALISATION OF EVENT RECONSTRUCTION

This chapter introduces the formal foundations of the SADFC approach. These foundations are composed of three elements: a formal definition of the event reconstruction problem, operators to carry out the reconstruction and a digital investigation process model. As said in Chapter 5, the introduction of these elements fulfils several objectives. First, the formal definitions of the crime scene and the elements composing it disambiguates the concepts handled by our tools. These definitions are, in addition, a first step in the conception of an ontology for the representation of digital incidents. Second, the formalisation of the event reconstruction operators also meets the legal requirements, including bringing credibility to the results produced. Indeed, if the tools are based on formal theories, it is possible to mathematically prove the veracity of the results produced by them. Third, the introduction of a precise investigation process model allows to explain clearly how the investigation is carried out, and, therefore, it ensures the reproducibility of evidence. Finally, this research work is intended to produce automated tools. As argued in Chapter 5, the introduction of a precise investigation process model contributes to the reproducibility of the evidence.

This chapter is structured as follows. After introducing the general notions of time and location, Section 6.1 defines the characteristics of a crime scene (and more particularly the digital crime scenes) and the entities composing it. As states in Chapter 5, the process of event reconstruction can be decomposed in two phases: the specification of events (extraction of the information from the crime scene and construction of a timeline representing the incident) and the analysis of the resulting timeline. To carry out this two-stage process, operators formally defined are introduced in Section 6.2. These reconstruction operators must be then integrated into the digital investigation process. Section 6.3 presents PaLADIN, a process model for digital investigation based on the state of the art proposed in Chapter 4. To help the reader in the consultation of this chapter, the following glossary lists all the symbols used in this chapter.

FORMALISATION

L Set of virtual locations.	E_i Set of illegal events.
H Set of hierarchical levels composing a path.	E_c Set of correlated events.
CS Set of crime scenes.	E_n Set of non related events.
PCS Set of physical crime scenes.	E_{inc} Set of events composing the incident.
DCS Set of digital crime scenes.	\leq_H Total order organising the hierarchical levels of a path.
N Set of entities.	γ_N Binds an entity to its characteristics.
E Set of events.	γ_O Binds an object to its characteristics.
O Set of objects.	γ_S Binds a subject to its characteristics.
S Set of subjects.	γ_E Binds an event to its characteristics.
F Set of footprints.	γ_F Binds a footprint to its characteristics.
C Set of characteristics.	σ_S Relations linking an event to a subject.
C_N Set of characteristics of entities.	σ_O Relations linking an event to an object.
C_E Set of characteristics of events.	σ_E Relations linking an event to an event.
C_O Set of characteristics of objects.	σ_F Relations linking a footprint to an entity.
C_S Set of characteristics of subjects.	
C_F Set of characteristics of footprints.	

6.1/ FORMAL DEFINITION OF CRIME SCENES

This section gives a definition of a crime scene. This implies the definition of the entities composing it, including the events taking place in it, the people and the processes which performs actions that affect the crime scene and the objects composing the scene. In addition, any entity constituting the crime scene can be located in time, in space or in time and space. These two last notions are also defined in this section.

6.1.1/ CRIME SCENE

In (Carrier et al., 2003b), the authors introduce the notion of digital crime scene as a part of a physical crime scene. They define a **physical crime scene** as a physical environment containing **physical evidence** related to an incident. The environment in which the initial physical criminal event takes place is called **primary physical crime scene**, while the subsequent scenes are called **secondary physical crime scene**. A **digital crime scene** is defined as a virtual environment created by hardware and software and containing **digital evidence** related to an incident. The environment in which the initial digital criminal event takes place is called **primary digital crime scene**, while the subsequent scenes are called **secondary digital crime scene**. A digital crime scene may be a computer, a phone or any electronic devices.

In our works, we define a **Crime Scene** CS as an environment in which an incident takes place and by $CS = \{PCS, DCS\}$ where:

- *PCS* is a set containing the **Physical Crime Scenes**. At the beginning of an investigation, *PCS* is initialised with the location where the incident takes place. However, in an investigation, the crime scene is not limited to only one building. Due to network communication, for example, the initial physical crime scene may be extended to a set of new physical crime scenes if one of the protagonists communicated with another person through the network, downloaded a file from a remote server, etc. In these cases, the seizure of the remote machines (and by extension, the creation of new physical crime scenes and digital crime scenes) should be taken into account as it may be relevant for the investigation.
- *DCS* is a set containing the **Digital Crime Scenes**. Unlike (Carrier et al., 2003b), there is no distinction between primary and secondary digital crime scene in our works to simplify the model.

There are several configurations of crime scenes involving digital objects. First, a crime scene can be simple or complex depending on whether the crime scene is composed of one (simple) or several (complex) scenes of physical crimes. Then, a crime scene can be connected or not. A crime scene is connected if communications are observed between digital objects contained in the scene. An example may be a case in which a person makes available defamatory documents which encourage violence against a person via a website. In this case, the physical crime scene is the room where the devices used to upload the documents to the server are. These devices may be the personal computer of the suspect, network devices or a phone and are considered as digital crime scenes. Since the case involves communications through a network (if it is a remote server and not a personal server located in the physical crime scene), it is then necessary to consider the physical place where the server is located (new physical crime scene). In this example, the crime scene is a complex scene composed of two physical crime scenes. This scene is connected because digital devices of the scenes communicate together.

6.1.2/ TIME AND LOCATION

Time and location are two notions extensively used during the investigations. During an investigation, investigators are working to collect enough information to locate accurately in time past events and locate precisely in space people and objects involved in the incident. To formalise the reconstruction of events, it is therefore first of all necessary to define the notion of time and location.

6.1.2.1/ TIME

In our works, we use two time notions: instant and interval. An instant is defined by a date and a time. The use of a time interval allows to represent the notion of uncertainty (Liebig et al., 1999). For example, when the start time of an event cannot be determined

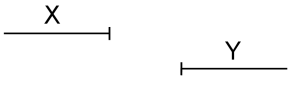
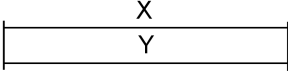
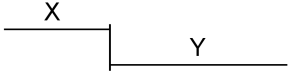
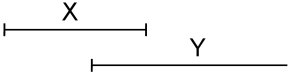
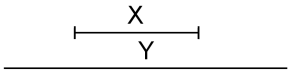
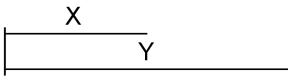
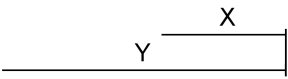
Functions	Example	Constraints
before(X,Y)		$x_{t_{end}} < y_{t_{start}}$
equals(X,Y)		$x_{t_{start}} = y_{t_{start}} \ \&\& \ x_{t_{end}} = y_{t_{end}}$
meets(X,Y)		$x_{t_{end}} = y_{t_{start}}$
overlaps(X,Y)		$x_{t_{start}} < y_{t_{start}} \ \&\& \ x_{t_{end}} > y_{t_{start}}$
during(X,Y)		$x_{t_{start}} > y_{t_{start}} \ \&\& \ x_{t_{end}} < y_{t_{end}}$
starts(X,Y)		$x_{t_{start}} = y_{t_{start}}$
finishes(X,Y)		$x_{t_{end}} = y_{t_{end}}$

Table 6.1: Allen algebra

accurately, the use of a time-interval allows to approximate it. The use of intervals requires the introduction of a specific algebra (e.g. to order events). In our work, the Allen algebra (Allen, 1983), illustrated in the columns Functions and Example of Table 6.1, is used. From this, we have defined a set of constraints (see column Constraints of Table 6.1) that events have to meet to validate or not the functions. We are aware of the problems caused by temporal heterogeneity and anti-forensics techniques on the quality and the accuracy of timestamps (granularity, timestamps offset and alteration, time zones, etc). Several works try to characterise the phenomena related to the use of time information in digital investigations and to propose techniques to solve these problems (Schatz et al., 2006), (Gladyshev and Patel, 2005), (Forte, 2004). In this thesis, we assume that all timestamps used in our model are adjusted and normalised beforehand by a process that will be the subject of future work (the proposed model is generic enough to incorporate future solutions). We consider that each function returns the value 1 if events meet the constraints and 0 otherwise.

6.1.2.2/ LOCATION

The location is a key concept in an investigation as it allows to locate the protagonists and the physical and the digital objects from the crime scene. In our work, we define two types of locations: physical location and virtual location. A physical location used

landmarks that can be used in the physical world to locate a physical object. A physical location can be used to locate a person using a postal address for example or a machine thanks to a MAC address or an IP address.

A virtual location is, for its part, used to locate a digital object using markers used in the digital world only. This location can be local, in case of a file stored locally (e.g. C:\Users\UserX\Downloads\contagioMalware.exe), or remote, in case of a resource stored on a server (<http://malwareWebsite.com/MalwareDL/contagio.html>). The virtual location $l \in L$ of an object defined the location of this object on a file system. It takes the form of a hierarchical path enabling to find the object. This path is made of hierarchical elements $h \in H$ separated by the symbols \ or /. Thus, a virtual location is defined as a vector describing the absolute path of the object, $l \in L = \langle h_r \in H | \leq_H \rangle$ where r is the rank (i.e. level) of the element in the hierarchy of the file system and \leq_H is a total order as $h_i \leq_H h_j$ if $i \leq j$. Elements of H can be of three types:

- A hostname giving the name of the machine on which the path is accessible: a hostname in case of a remote virtual location, a drive letter in case of a local virtual location.
- A folder which give information about the location of the object in the file system.
- The name of the object.

For example, the path C:\Users\UserX\Downloads\contagioMalware.exe is made of a drive letter C:, a succession of folders Users, UserX, Download and a name contagioMalware.exe. As a second example, the URL <http://malwareWebsite.com/MalwareDL/contagio.html> is composed of a hostname <http://malwareWebsite.com>, a folder MalwareDL and a name contagio.html.

6.1.3/ ENTITIES COMPOSING A CRIME SCENE

Multiple entities make a crime scene. In our work we argue that the concept of entity encompasses the concepts of events (actions occurring in the crime scene), objects (resources interacting with events such as a knife, a book, a digital file, etc.), subjects involved in events and footprints left on the crime scene as $N = E \cup O \cup S \cup F$ where N is the set of all entities of the crime scene, E is the set of events occurring in the crime scene, O is the set of objects interacting with events from E , S is the set of subjects involved in events from E and F is the set of footprints retrieved in the crime scene. Given the set containing all attributes $C_N = C_O \cup C_E \cup C_S \cup C_F \subseteq C$ (defined later in this section), each entity is described by a set of attributes as $x \in N = \{c \in C_N | \forall \gamma_N \in \{\gamma_O, \gamma_S, \gamma_E, \gamma_F\}, x\gamma_N c\}$ where γ_N is the relation linking an entity with its attributes.

6.1.3.1/ EVENT

A crime scene is a space where a set of events $E = \{e_1, e_2, \dots, e_i\}$ takes place. An event may be the drafting of a document, the reading of a webpage or a conversation via instant messaging software. An event takes place in a time-interval defined by a start time and an end time. These boundaries define the life cycle of the event. An **event** $e \in E$ is defined by $e = \{t_{start}, t_{end}, l_e, S_e, O_e, E_e\}$ where:

- t_{start} is the start time of the event, t_{end} is the end-time of the event.
- l_e is the physical location where the event took place. This location may be information about a machine (i.e. IP, MAC address, etc.) and the postal address where it is located.
- S_e is a set containing all subjects involved in the event. $S_e = \{s \in S \mid e \in E, s \sigma_S e\}$ where σ_S is a composed relation used to link an event $e \in E$ with a subject $s \in S$. The relation σ_S is defined below.
- O_e is a set containing all objects related to the event e . $O_e = \{o \in O \mid e \in E, e \sigma_O o\}$ where σ_O is a composed relation used to link an event $e \in E$ with an object $o \in O$. The relation σ_O is defined below.
- E_e is the set containing all events with which the event is correlated. $E_e = \{x \in E \mid e \in E, e \sigma_E x\}$ where σ_E is a composed relation used to link an event $e \in E$ with an event $x \in E$. The relation σ_E is defined below.

σ_E is composed of relations used to link two events $x, e \in E$ and can be defined in the following way $\sigma_E = x \text{ composes } e \vee e \text{ composes } x \vee x \text{ causes } e \vee e \text{ causes } x$. In our works, $x \text{ isCorrelated } e$ means that x is linked to e on the basis of multiple criteria: use of common resources, participation of a common person or process, temporal positions of events. We distinguish two special cases of the relation of correlation:

- **Composition relationship:** $x \text{ composes } e$ means that x is an event composing e . For example, an event representing a Windows session is composed of all events initiated by the user during this session. Let $x = \{t_{xstart}, t_{xend}, S_x, O_x, E_x\}$ be an event composing $e = \{t_{estart}, t_{eend}, S_e, O_e, E_e\}$, the relation of composition implies a set of constraints. First, a temporal constraint requiring that sub-events take place during the parent event. Using Allen relations, if $x \text{ composes } e$ then $equal(x, e)$ or $during(x, e)$ or $starts(x, e)$ or $finishes(x, e)$. Sub-events have also constraints on participating subjects as well as the objects with which the event interacts. If $x \text{ composes } e$ then $S_x \subseteq S_e$ and $O_x \subseteq O_e$. Thus, $x \text{ composes } e = [equal(x, e) \vee during(x, e) \vee starts(x, e) \vee finishes(x, e)] \wedge (S_x \subseteq S_e) \wedge (O_x \subseteq O_e)$.
- **Causality relationship:** $x \text{ causes } e$ means that x has to happen to allow e to happen. For example, an event describing the download of a file from a server is caused by the event describing the connection to this server. An event can have several causes and can be the cause of several events. Let $e = \{t_{estart}, t_{eend}, S_e, O_e, E_e\}$ be an event caused by $x = \{t_{xstart}, t_{xend}, S_x, O_x, E_x\}$, the relation of causality implies a temporal

constraint requiring that the cause must happen before the consequence. Using Allen algebra, $x \text{ causes } e = [before(x, e) \vee meets(x, e) \vee overlaps(x, e) \vee starts(x, e)] \vee (S_x \cap S_e) \vee (O_x \cap O_e)$.

6.1.3.2/ SUBJECT

During its life cycle, an event involves **subjects**. Let S be the set containing subjects covering human actors and processes (e.g. Firefox web browser, Windows operating system, etc.), a subject $x \in S$ corresponds to an entity involved in one or several events $e \in E$ and is defined by $x = \{c \in C_S \mid x \gamma_S c\}$ where:

- C_S is a set containing all the attributes which can be used to describe a subject. Such an attribute may be the first name and the last name of a person, the identifier of a web session, the name of a Windows session, etc.
- γ_S is the relation used to link a subject with the attributes of C_S describing it.

The relations σ_S linking an event $e \in E$ with a subject $s \in S$ are composed of two types of relations to link and can be defined in the following way $\sigma_S = s \text{ isInvolved } e \vee s \text{ undergoes } e$:

- **Participation relationship:** $s \text{ isInvolved } e$ means that s initiated or was involved in e . For example, the user of a computer is involved in an event representing the login to the session, etc.
- **Repercussion relationship:** $s \text{ undergoes } e$ means that s is affected by the execution of e . For example, a user is affected by the removal of one of his files, etc.

6.1.3.3/ OBJECT

During its life cycle, an event can also interact with **objects**. An object may be a webpage, a file or a registry key for example. An object $x \in O$ is defined by $x = \{c \in C_O \mid x \gamma_O c\}$ where:

- C_O is a set containing all the attributes which can be used to describe an object. Such an attribute may be, for example, the virtual location of an object ($l \in L \subseteq C_O$), its size, etc.
- γ_O is the relation used to link an object with the attributes of C_O describing it.
- $O \subseteq \wp(C_O)$ is the set of the objects, meaning that $o \in O$ belongs to the power set of C_O . An object is a composition of one or several attributes of C_O .

Note, that for easier human understanding, an object can also be seen as a composition of attributes and objects (because objects are sets of attributes), e.g. a registry key is an object made of several attributes such as its value and its key name. This registry key is also an attribute of the object representing the database containing all keys of the system. The relations σ_O linking an event $e \in E$ with an object $o \in O$ are composed of four

types of relations and can be defined in the following way $\sigma_O = e \text{ creates } o \vee e \text{ removes } o \vee e \text{ modifies } o \vee e \text{ uses } o$:

- **Creation relationship:** $e \text{ creates } o$ means that o does not exist before the execution of e and that o is created by e .
- **Suppression relationship:** $e \text{ removes } o$ means that o does not exist anymore after the execution of e and that o is deleted by e .
- **Modification relationship:** $e \text{ modifies } o$ means that one or more attributes of o are modified during the execution of e .
- **Usage relationship:** $e \text{ uses } o$ means that one or more attributes of o are used by e to carry out its task.

6.2/ FORMAL OPERATORS FOR EVENT RECONSTRUCTION

As said in Chapter 3, the event reconstruction is intended to move from a static crime scene containing traces of past events to a timeline describing events that happened in the past. The final goal of the event reconstruction is to draw conclusions using information contained in the produced timeline. To do this, the event reconstruction process can be decomposed in two phases. First, investigators have to reconstruct the timeline by identifying, using the footprints left in the crime scene, the past events and the entities that have interacted with them. In our work, this phase is called **Specification of Events** and is discussed in Section 6.2.1. The second phase is to interpret and analyse the timeline to understand what happened in the past and determine the responsibilities of each protagonist. This phase is called "Timeline Analysis and Incident Scenario Extraction" and is discussed in Section 6.2.2.

6.2.1/ SPECIFICATION OF EVENTS

Each event is made to achieve an action (e.g. removing a file, modify a register key, load a webpage, etc.). The events taking place in a digital crime scene can be classified as follows: $E_{CS} = \{E_{iCS} \cup E_{cCS} \cup E_{nCS}\}$. For easier notation, we write here $E = \{E_i \cup E_c \cup E_n\}$ where:

- E_i is a set containing **Illicit** events. This set contains all actions considered as infractions by the laws. For example, an event representing the upload of defamatory documents to a website is an event of E_i .
- E_c is a set containing the events **Correlated** to the incident as $E_c = \{e \in E \mid e \sigma_E x, x \in E_i\}$. This set contains all legal events which are linked with a set of illicit events x .
- $E_n = \{E \setminus (E_i \cup E_c)\}$ is a set containing the events which are **Not relevant** for the investigation.

According to (Ribaux, 2013), a **footprint** is the sign of a past activity and a piece of information allowing to reconstruct past events. A footprint may be a log entry or a web history, for example, as a log entry gives information about software activities and web histories provide information about the user's behaviour on the Web. Let F be a set containing all footprints related to a case, a footprint $x \in F$ is defined by $x = \{c \in C_F \mid x \gamma_F c\}$ where:

- C_F is a set containing all the information carried out by a footprint. For example, a log entry of a web browser about the visit of the webpage may contain the title and the URL of the webpage in addition to the date of the visit and the session number of the user.
- γ_F is the relation used to link a footprint with the attributes of C_f used to describe it.

The relation σ_F is used to link a footprint $f \in F$ with an entity $en \in \{E \times O \times S\}$. This relation is called **Support relationship**: f supports en means that f is used to deduce one or more attributes of en . We define a function *support* which can be used to know the footprints used to deduce a given entity: $support(en \in \{E \times O \times S\}) = \{f \in F \mid f \sigma_f en\}$. Footprints are the only available information to define past events and can be used by investigators to reconstruct the events which happened during an incident. However, the imperfect and incomplete nature of the footprints can lead to produce approximate results. It is therefore not always possible to determine which event is associated with a given footprint. In addition, it is not always possible to fully reconstruct an event from a footprint. Thus, a footprint can be used to identify one or several features:

- The temporal features or the location of an event. For example, each entry of a log (that can be considered as a footprint) provides temporal information to establish the time at which the action occurred. Entries may also provide information about the location of an event as the name of the machine or an IP address.
- A relation between an event and an object. For example, an entry in a log written by the file system may give information about the modification of a file. Therefore, a link can be established before the event of modification and the file.
- A relation between an event and a subject. For example, all footprints produced by the web browser Firefox are stored in a folder named with the profile name of the user. This allows to link each Firefox event to the user designated by this name.
- The features of an object. For example, a footprint extracted from web browser logs can be used to determine the URL and the title of a webpage.
- The features of a subject. For example, logs of OS can be used to determine the session identifier of a user.

Since investigators have ensured the preservation of the crime scene, it becomes a protected static environment containing a set of footprints. After the collection of all the footprints of the crime scene, the goal of the specification of events consists in moving from the static crime scene to a timeline describing the dynamics of the events which happened in the past. Describing the events means identify all the events E using foot-

prints of F . To carry out the specification of events, we introduce three types of operators. The goal of **extraction operators** is to identify and extract relevant information contained in digital footprints from various sources. Sources are chosen according to the definition of the perimeter of the crime scene. The relevancy or the irrelevancy of information contained in footprints is determined by the investigators, according to the goals of the investigation. For example, in a case involving illegal downloads of files, the investigator will pay more attention to information related to the user behaviour on the web while the logs of word processing software will be ignored. The **mapping operators** create entities (events, objects and subjects) associated to the extracted footprints. These operators take the form of mapping rules allowing to connect attributes extracted from footprints to attributes of events, objects and subjects. A large part of the features of an event can be determined by extraction operators from footprints collected in the crime scene. However, the identification of some kinds of features requires the use of advanced techniques such as inference. The **inference operators** allow to deduce new knowledge about entities from existing knowledge. Unlike **extraction operators** which use knowledge of footprints, inference operators use the knowledge about events, objects and subjects (knowledge generated by mapping operators). The aim of these operators is to improve the knowledge we have about the past in order to enhance the efficiency of the subsequent analysis. Indeed, the more investigators know, the more they are able to reach accurate and true conclusions.

6.2.2/ TIMELINE ANALYSIS AND INCIDENT SCENARIO EXTRACTION

After the specification of events, the analysis of the timeline identifies the scenario of the incident. Identifying an incident means identify all the events $E_{inc} = E_i \cup E_c$ using footprints of E where E_{inc} is a set containing all the events that are directly **related to the INCIDENT**. Events ordered chronologically describing an incident are called the **scenario of the incident**. The **analysis operators** are used to help the investigators during the interpretation of the timeline and the reconstruction of the scenario of the incident. These operators are used to identify relations between events and to highlight the relevant information contained in the timeline. At this stage of progress of the project, we propose a single analysis tool which is used to identify correlations between events. Another important aspect of the analysis is the identification of illicit events. However, it was not possible to introduce such a tool in the time allocated for this thesis. Thus, this step has to be carried out manually by the user. However, the development of this tool is a future work.

The first analysis tool proposed in our approach is a process allowing to detect correlation between a pair of events. The correlation is admitted as a relationship with a broad semantic that cover causal relationships and other semantic links. The identification of such relationships is particularly relevant for the investigators as it allows them to quickly

have an overall view of the events composing an incident and the links between them. The correlation between two events $e, x \in E$ is measured by the following function:

$$Corr(e, x) = \begin{cases} \text{If } Corr_{KBR}(e, x) = 1 \text{ then } 1 \\ \text{else } \frac{\alpha * Corr_T(e, x) + \beta * Corr_S(e, x) + \gamma * Corr_O(e, x)}{3} \end{cases} \quad (6.1)$$

$Corr_T(e, x)$, $Corr_S(e, x)$ and $Corr_O(e, x)$ are normalised to get a value between 0 and 1. $Corr_T(e, x)$, $Corr_S(e, x)$ and $Corr_O(e, x)$ can be weighted to allow to give more importance to one of the correlation functions using α , β and γ factors. If $Corr_{KBR}(e, x) = 1$ which means that a knowledge-based rule is satisfied, then $Corr(e, x) = 1$. Unlike other factors of correlation, $Corr_{KBR}(e, x)$ is based on knowledge defined by experts. As this knowledge is reliable, when a rule of the set is satisfied, it can be considered that the two events are correlated. $Corr(e, x)$ can also be ordered and threshold to deal with data volume constraints by selecting the most significant correlations. Those four correlations are described in the following way:

Temporal Correlation, $Corr_T(e, x)$: First of all, a set of assumptions about the temporal aspect is defined (according to the Allen algebra given in Table 6.1):

- The greater the relative difference between the two events $before(e, x)$ is, the lower the temporal relatedness is and reciprocally.
- The temporal relatedness is maximal (equals 1) for functions $meets(e, x)$, $overlaps(e, x)$, $during(e, x)$, $finishes(e, x)$, $starts(e, x)$ and $equals(e, x)$.

The temporal correlation between two events $e, x \in E$ is measured by the following function:

$$\begin{aligned} \text{Thus, } Corr_T(e, x) = & starts(e, x) + equals(e, x) + meets(e, x) \\ & + overlaps(e, x) + during(e, x) + finishes(e, x) + before(e, x) \end{aligned} \quad (6.2)$$

where $starts(e, x)$, $equals(e, x)$, $meets(e, x)$, $overlaps(e, x)$, $during(e, x)$, $finishes(e, x)$ are binary functions and $before(e, x) = \frac{1}{(x_{t_{start}} - e_{t_{end}})}$. Previous assumptions state that the more two events are close in time, the more it is likely that these events are correlated. Because of time granularities, and multi-tasks computers, if two events start at the same time, the relatedness is maximal.

Subject Correlation, $Corr_S(e, x)$: This score quantifies the correlation between two events regarding subjects involved in each event. The following hypothesis are defined according to the core idea of the field of data mining. For example, the formal concept analysis (Ganter et al., 1997) groups objects regarding to the attributes they share. In the same way, in statistics, the principal component analysis groups observations measuring how far they are spread (variance). The relatedness between e and x increases

proportionally to the number of common subjects they share regarding to relations of **participation** and **repercussion**. The subject correlation between two events $e, x \in E$ is computed using the following function:

$$Corr_S(e, x) = |S_e \cap S_x| / \max(|S_e|, |S_x|) \quad (6.3)$$

In order to quantify the importance of this similarity, the numerator in the formula is divided by the number of subjects interacting with the event which interacts with the largest number of subjects. Therefore, if the number of similar subjects between the two events is significant relatively to the number of subjects interacting with them, then, the correlation score is high. If two events have a few number of similar subjects compared to the number of subjects interacting with one of them, the correlation score is low.

Object Correlation, $Correlation_O(e, x)$: This score quantifies the correlation between two events regarding objects used, generated, modified or removed by the events. The following hypothesis is based on the same core idea than the subject correlation. The relatedness between e and x increases when they interact with a similar object regarding to relations of **creation**, **suppression**, **modification** and **usage**. We do not consider only common objects but also objects that have similar virtual locations. Therefore, the object correlation score increases if two events interact with a same object, or if two events interact with two different objects which have nearby locations. For example, if two events interact with two different objects both located in `C:\ProgramFiles(x86)\Adobe\`, we can admit that the two events are related as they both interact with objects related to Adobe tools. Given two objects $o_1, o_2 \in O$ virtually located in $l_{o1}, l_{o2} \in L$, the distance between these objects is given by the following formula:

$$d(o_1, o_2) = \frac{2 * |l_{o1} \cap_H l_{o2}|}{|l_{o1}| + |l_{o2}|} \quad (6.4)$$

The intersection $l_1 \cap_H l_2$ between two virtual locations $l_1, l_2 \in L$ is defined as follows:

$$l_1 \cap_H l_2 = \{ h_{l_1 r} \in H \mid \leq_H r \in [0; \max_i(h_{l_1 i}, h_{l_2 i}) \mid \nexists h_{l_1 j} \neq h_{l_2 j}, j \leq i] \} \quad (6.5)$$

Let four objects $o_1, o_2, o_3 \in O$ respectively associated with the locations $vl_1 \in L = \langle C:, Users, UserX, Documents, setup.exe \rangle$, $vl_2 \in L = \langle C:, Users, UserX, Downloads, Reports, report.pdf \rangle$, $vl_3 \in L = \langle C:, Users, UserX, Downloads, Reports, meeting.pdf \rangle$ and $vl_4 \in L = \langle C:, Users, UserY, Documents, driver.exe \rangle$, the distance between all the possible pair of objects are: $d(o_1, o_2) = \frac{2*3}{5+6} = 0.54$, $d(o_1, o_3) = \frac{2*3}{5+6} = 0.54$, $d(o_1, o_4) = \frac{2*2}{5+5} = 0.4$, $d(o_2, o_3) = \frac{2*5}{6+6} = 0.83$, $d(o_2, o_4) = \frac{2*2}{6+5} = 0.36$, $d(o_3, o_4) = \frac{2*2}{6+5} = 0.36$. The most similar pair of objects is thus (o_2, o_3) as their paths are the most similar among the pairs of objects studied. Using the distance between objects, the object correlation between two events

$e, x \in E$ is computed using the following formula:

$$Corr_O(e, x) = \max(d(O_{ei}, O_{xj})), i \in [1, n], j \in [1, m] \quad (6.6)$$

with $n = |O_e|$, the number of objects interacting with the event e and $m = |O_x|$ the number of objects interacting with the event x .

Rule-based Correlation, $Corr_{KBR}(e, x)$: In addition to the previous factors (time, subject, object), rules based on expert knowledge can be used to correlate events. The rule-based correlation between two events $e, x \in E$ is measured by the following function:

$$Corr_{KBR}(e, x) = \begin{cases} \text{If } \sum_{r=1}^n rule_r(e, x) > 0 \text{ then } 1 \\ \text{else } 0 \end{cases} \quad (6.7)$$

with $rule_r(e, x) = 1$ if the rule is satisfied and 0 otherwise.

This section introduces the operators required to carry out the event reconstruction, from the extraction of footprints to the analysis of the events. The next section introduces a new digital investigation process model integrating our event reconstruction operators in the larger process that is the digital investigation.

6.3/ PROCESS MODEL FOR THE AUTOMATION OF DIGITAL INVESTIGATIONS

To be able to process the quantity of data found in a crime scene, the investigators need the assistance of digital forensics software tools to automate parts of the investigative process. In addition, the investigators have to satisfy the requirements of justice to make admissible all results produced during the investigation. To provide clear explanations about the evidence found and conclusions reached during the investigation, a formal and standard definition of the investigation process is needed. This definition allows to ensure the reproducibility of the process and give credibility to results by explaining to the court the process used to get the results. To answer the limits highlighted in Chapter 4, this section introduces a new investigation process model which is highly-detailed and complete enough to meet legal requirements and to be used as a framework for the development of automated digital forensics tools.

In the literature, numerous digital investigation process models have been proposed. However, as (Beebe and Clark, 2005) argue "existing models are high order process models that focus on the abstract, rather than the more concrete principles of the investigation". The first goal of the definition of this new process model is to go one step further in the accuracy and the completeness of the process model to support the automation of

investigative tools. Indeed, our process model should be concrete and accurate enough to be translatable in algorithms. It is especially important to detail the steps inherent to the digital investigation phases and especially the event reconstruction which is the focus of this thesis. For this, we try to follow the example of (Stephenson, 2003), (Carrier and Spafford, 2004a) and (Carrier and Spafford, 2004b) that are two models with a detailed view of this part of the investigation. The aim of our model is to define the various phases of the digital investigation process: their types, the order between them and the data flow through the whole process.

The introduction of a digital investigation process must be integrated into a larger picture encompassing also the physical investigation process. As in (Carrier et al., 2003b), our goal is to introduce a complete and standalone model encompassing the physical aspects and the digital aspects of the investigation in order to introduce a complete view of the investigation process. Indeed, to complete a digital investigation, investigators need to collect digital devices in physical crime scenes. The proper conduct of the phases composing the physical investigation is essential to ensure the smooth running of the subsequent digital investigation. Therefore, the process model presented in this section provides a broad view of the investigation, covering both the physical and digital aspects of it.

This model, called PaLADIN, is based on theoretical foundations introduced previously in this chapter. In addition, to enable a broad adoption of this model in the digital forensics community, this model is meant to be a synthesis of the existing process models studied in Chapter 4. Our goal is to introduce a new model that incorporates the strengths of existing models. For this purpose, we try to highlight the steps represented in a majority of existing models while taking into account the innovative ideas proposed by each model when deemed relevant.

6.3.1/ SPECIFICATION OF PALADIN

The proposed model is divided in three parts: **pre-investigation**, **investigation core** and **post-investigation**. The steps composing PaLADIN are described below. It should be noted that PaLADIN contains many steps that are already mentioned in the state of the art given in Chapter 4. The readers are invited to refer to this chapter for more information on the nature of these steps.

This model is designed to allow the automation of parts of it. In the next chapter, we introduce an ontology-centred architecture, named ANNALIST, which is meant to carry out automatically all steps in bold dashed boxes in Figure 6.1 (some parts are not yet implemented and will be further work). It implements the two phases of event reconstruction. The first phase is called "specification of events". This phase is carried out for each digital device found in every physical crime scene. It characterises the events happening

in each device. The second phase is called "timeline analysis". This phase is intended to analyse the timeline containing all the events identified during the previous phase. There is only one occurrence of this phase conducted for the entire survey as this phase aims to get an overview of the whole case. The operation of the process model PaLADIN is illustrated in the next section and in Chapter 9.

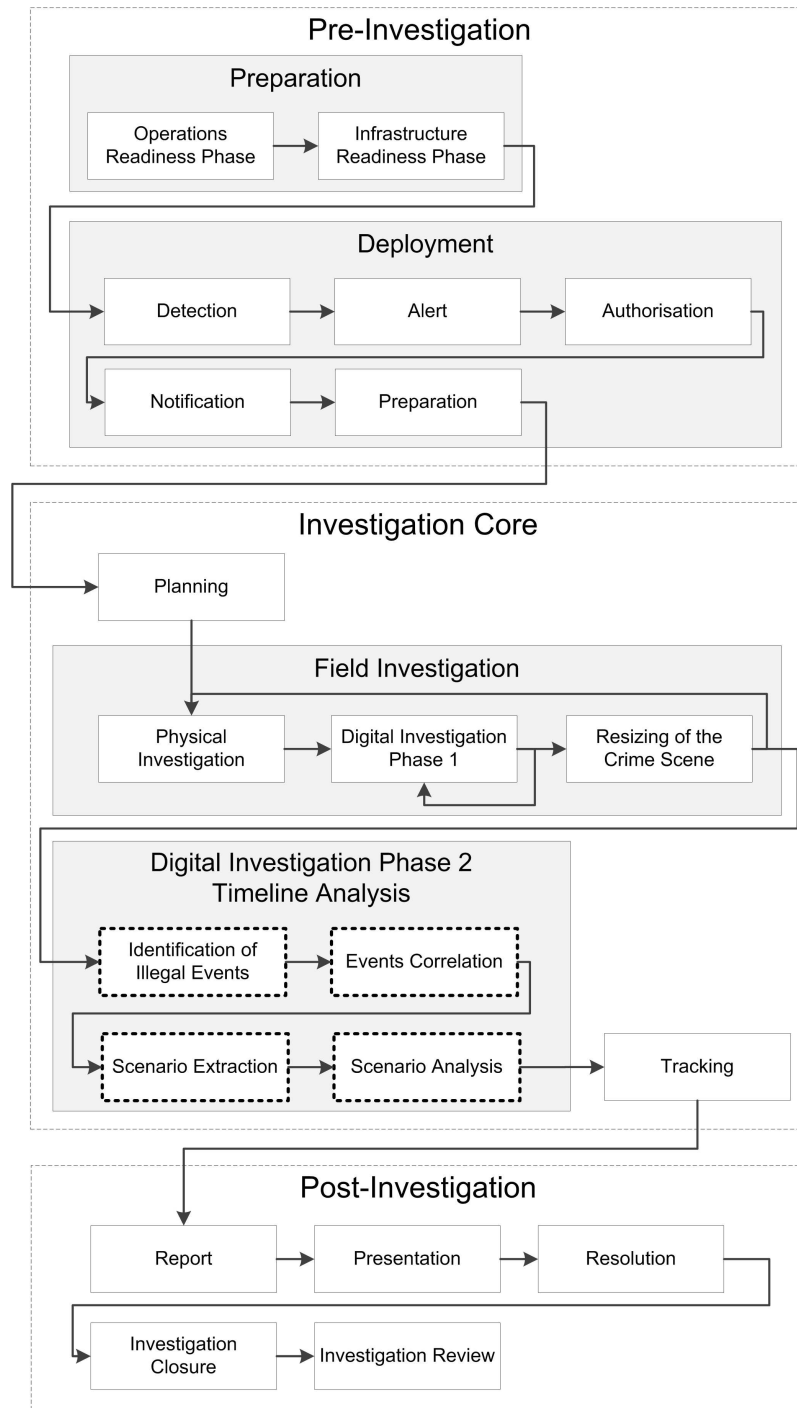


Figure 6.1: A complete view of PaLADIN

6.3.1.1/ PRE-INVESTIGATION

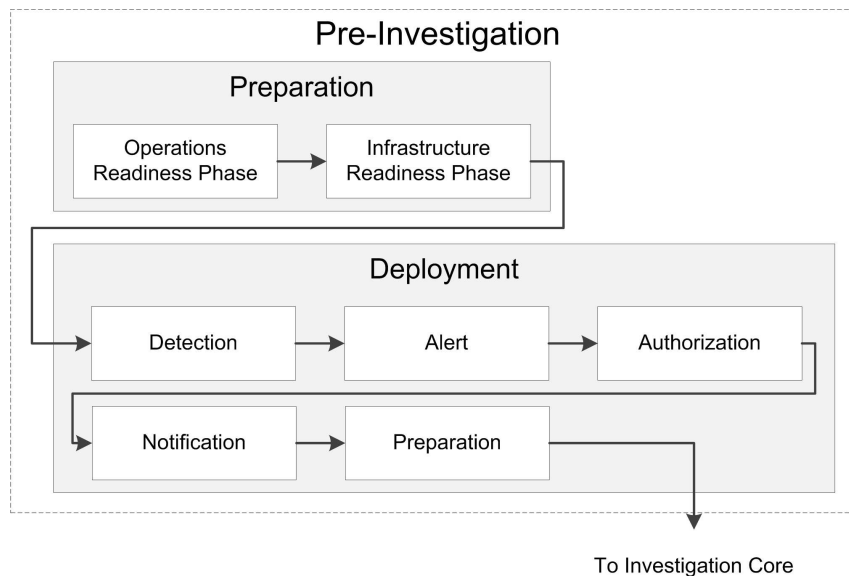


Figure 6.2: Pre-investigation

The pre-investigation, illustrated in Figure 6.2 is composed of two phases: **preparation** and **deployment**. The preparation intervenes before the incident happens and is made of two steps which are **operations readiness** and **infrastructure readiness**. These two steps occur before the detection of an incident and ensure that the investigative team is ready for any and all possibilities. They aim to prepare people composing the investigative team and the tools used by them and to ensure that mechanisms are in place to ensure that enough information is collected upstream of the incident.

The deployment starts after the incident happens and includes five phases. First, the **detection** is the identification of the incident by automated systems or witnesses in the crime scene. An alert is then sent to the appropriate organisation using phones or automatic messages during the **alert** step. This step also aims to avoid false alarms by confirming or not the incident using the information contained in the alert. After receiving the alert, the first responder acquires authorisation to investigate the crime scene during the **authorisation** step. The two last steps of the pre-investigation are the **notification** and the **preparation**. These steps prepare the arrival of investigators on the crime scene by preparing tools and equipments needed to carry out the investigation and to notify the persons concerned that an investigation is about to start.

6.3.1.2/ INVESTIGATION CORE

This part of the investigation starts when the investigators are arrived in the crime scene. An investigation in a crime scene made of physical crime scenes and digital crime scenes is a complex process. (Carrier et al., 2003b) define an investigation as a process starting

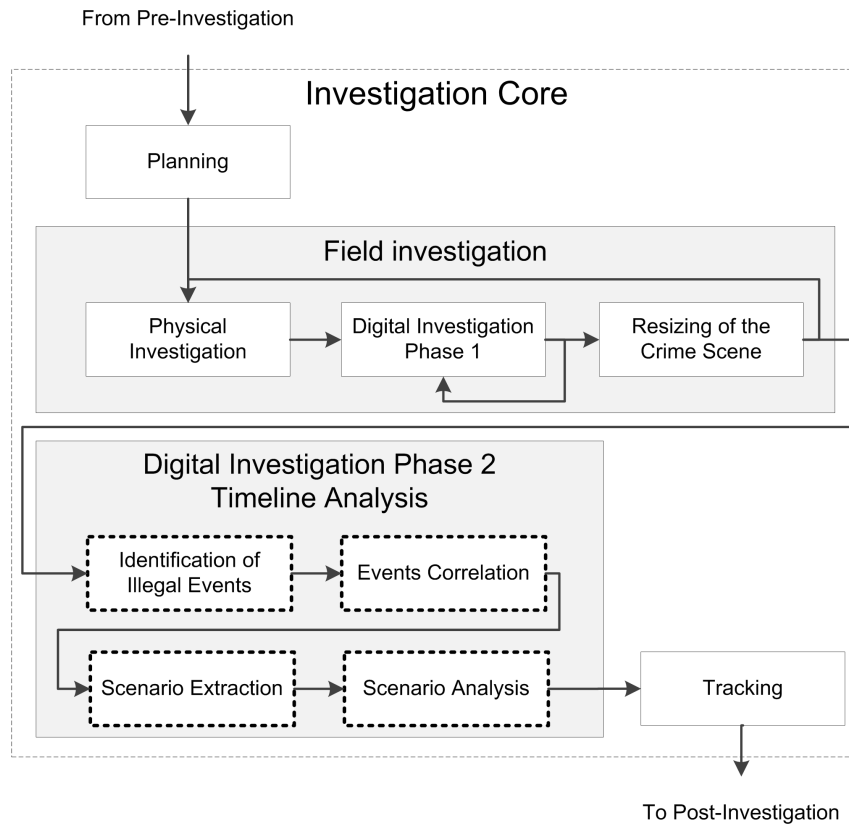


Figure 6.3: Investigation core

with a physical investigation of the physical crime scene. During the collection phase of the physical investigation, evidence containing digital footprints may be discovered. When such evidence is discovered, a digital investigation is carried out on this evidence to collect digital footprints contained on it. In our works, we define the investigation process using this approach.

The first step of the investigation core is the **planning** step. During it, the investigators choose a strategy to carry out the investigation efficiently. They determine the objectives of it, the systems that need to be investigated and the order in which they have to be studied. In addition, this phase also sizes the crime scene. Initially the crime scene CS is composed of the physical crime scene pcs where the incident takes places. This scene is added to the set PCS containing the physical crime scenes, $PCS = \{pcs\}$. Thus, $CS = \{\{pcs\}, \emptyset\}$. After the planning, the **field investigation** starts. This phase lasts until all scenes of PCS have been studied. The first step of this phase is the **physical investigation** of the next unstudied physical crime scene contained in PCS is carried out. During the physical investigation (defined below), digital devices are collected and added into DCS . A digital investigation is then carried out in every unstudied scene contained in DCS during the **digital investigation - phase 1**. Finally, each iteration of the field investigation ends with the **resizing of the crime scene**. In some cases, it may be needed to expand the crime scene to other scenes. If a digital device obj discovered in a physical

crime scene $pcs1 \in PCS$ is connected to a digital device which belongs to a physical space $ps \notin PCS$, then ps is added to PCS and obj is added to DCS . A new physical crime scene may also be added in other cases (e.g. if a protagonist moved to another place). If the size of the crime scene is modified during this step, it is then necessary to investigate the new scenes. A loop is used in the process model to continue to iterate through the field investigation phase while there are scenes in PCS to investigate.

The next phase is the **digital Investigation - phase 2** which aims to analyse the timeline and to extract the scenario of the incident. This phase starts with the **identification of illicit events**. The goal of this step is to identify the events belonging to the intersection $E_{ir} = E_i \cap E_r$, where E_r is a set containing the events which can be reconstructed and E_{ir} is the set of the illicit events which have been reconstructed. The identification of illicit events is a complex task that can be carried out by the investigator and/or by an operator able to identify them. This step is, at the moment, carried out manually by the investigators but we plan to propose an automated tool for the identification of illicit events in future work. The **events correlation** step then identifies the set E_{cr} which contains the events of E_r which are correlated to the events of E_{ir} . Correlations between events are identified automatically using the event correlation operator introduced in Section 6.2.2. The next step is the **extraction of the scenario of the incident**. This step extracts the sets E_{ir} and E_{cr} to get a set of events forming the scenario. The scenario is then analysed during the **scenario analysis**. This step is intended to interpret the scenario to establish the responsibilities of protagonists, identify culprits and the methods used to cause the incident. At the end of this step, investigators may be able to refute or confirm allegations about the incident and draw conclusions to present to justice. Once the suspects are identified, the **tracking** step aims to apprehend suspects and to present them to justice.

The steps composing the field investigation are now going to be presented. First, a physical investigation is a process made of the following seven steps (illustrated in Figure 6.4). First, the **protection of the physical crime scene** is intended to preserve the integrity and the state of the physical objects composing the scene. Then, the **documentation of the physical crime scene** and the **collection** step aim to document the crime scene and then search and collect the physical objects in the crime scene which may be used as evidence. When a physical evidence is a digital device which may contain digital evidence, the object (e.g. a computer) is added to DCS , the set of digital crime scenes. The **seizure**, the **transport**, the **storage** and the **evidence Documentation** are intended to transport the objects collected in the crime scene to a location more suitable for further investigation. During these four steps, the evidences are packed and documented in order to ensure their integrity and keep information about each evidence and the way they were collected.

The first phase of the digital investigation is a process made of the following steps (illustrated in Figure 6.5). The step called **live collection** is intended to collect volatile footprints of F and determine the associated events of E . It should be noted that if a

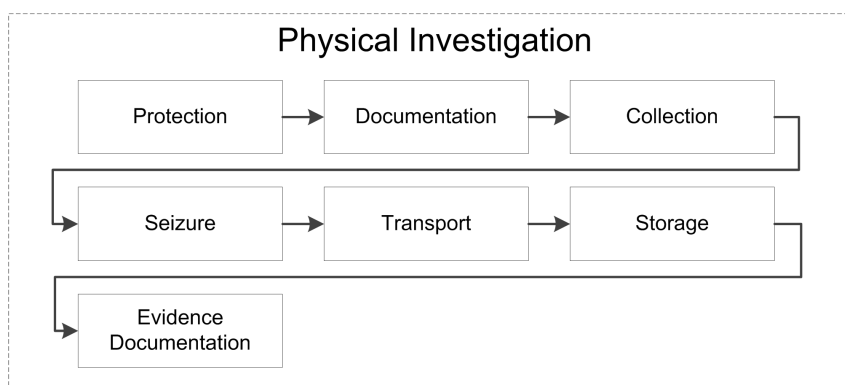


Figure 6.4: Physical investigation

live collection is needed for a digital device, then, it has to be done before the phases of collection, seizure, transport and storage of the physical investigation. Indeed, the live collection implies to not shut down the digital device to study. The digital scene is then protected and documented during the **protection of the digital crime scene** and the **documentation of the digital crime scene**. The next step is the **specification of events** which is itself composed of several steps described below. This step is discussed in more details in Section 6.2.1. First, the **extraction** collects footprints of F left on the digital crime scene. This step is carried out using approved software and specialised algorithms allowing to recover hidden or erased data (Palmer, 2001). Then, the **mapping** step extracts relevant information contained in footprints in order to reconstruct events of E accordingly. This step also reduces the amount of data to process by using filters. The output of this step is added to the set $E_r \subseteq E$ containing the events which can be recovered using the information left on the different digital crime scenes. A large part of the features of the events can be determined from footprints collected in the crime scene. However, the identification of some kinds of features requires the use of advanced techniques such as inference. Finally, the **knowledge enhancement** enriches the knowledge about events using the inference operators and add them to the set E_r . This step deduces new knowledge from the knowledge generated by the mapping step. These operators improve the knowledge we have about the past in order to enhance the efficiency of the subsequent analysis. Indeed, the more investigators know, the more they are able to reach accurate and true conclusions.

6.3.1.3/ POST-INVESTIGATION

The post-investigation is made of five steps ending the investigation: **report, presentation, resolution, investigation closure, investigation review**. This phase is illustrated in Figure 6.6.

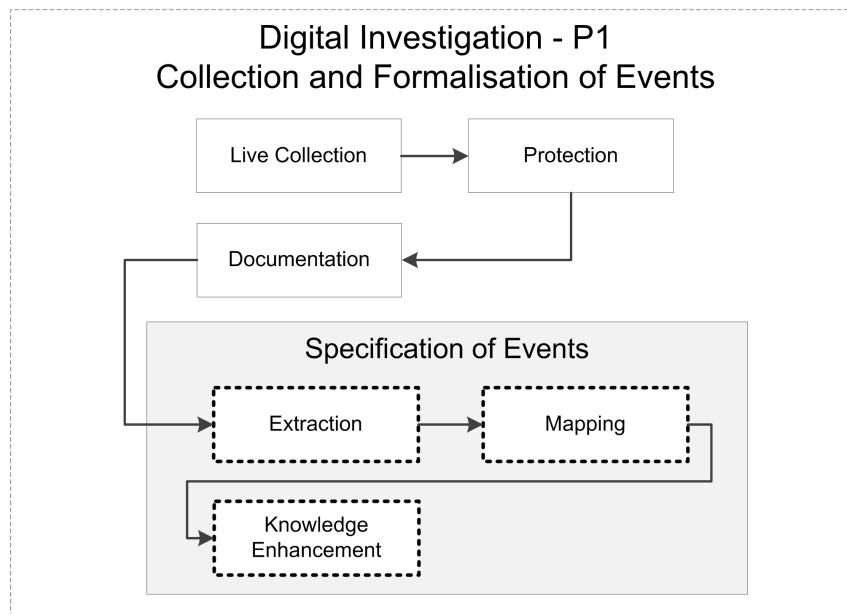


Figure 6.5: Phase 1 of the digital investigation

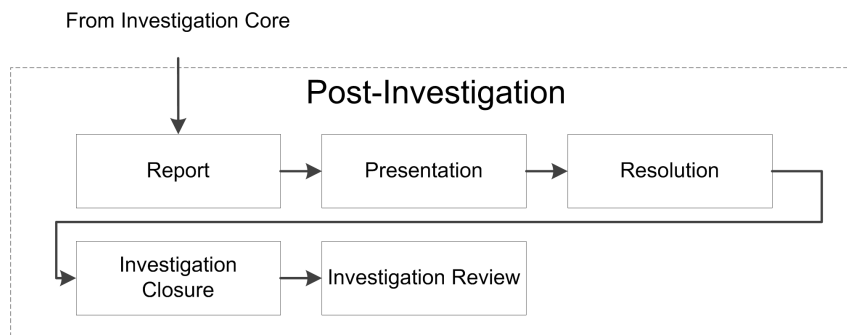


Figure 6.6: Post investigation

6.4/ CASE STUDY

This case study provides an example of the use of the formalisation to firstly, clarify it and secondly, present the functioning of the formal operators of reconstruction. This section also presents how the process model PaLADIN works and how it allows to structure the conduct of an investigation. It should be noted that this section is not intended to give technical details about how each step works. For more details, the readers can refer to Chapter 8 describing the software architecture. The proposed case study is a fictitious scenario involving a private investigator. The latter was approached by a company to conduct an investigation involving one of its employees. The head of the company suspects his employee to use the company's Internet connection to illegally download media content such as movies and TV shows. For simplification, this case study starts when the investigator arrived at the scene of the investigation. It takes place at the beginning of the investigation core, after the investigator has been alerted by the employer and after the

preparation of his equipment and software tools.

The first step of the investigation is the **planning** step. In this case, the investigator determines that the main objective of the investigation is to establish whether or not the employee illegally downloaded a file using the Internet connection of the company. He then sizes the crime scene and he establishes a list of objects potentially relevant to reach this objective. The crime scene is limited only to the office of the employee, which is isolated from the rest of the offices of the company. This office contains several digital equipments including the laptop used by the employee to work. This computer is the subject of special attention from the investigator as it is probably the equipment used to perform the download. The crime scene *CS* is therefore composed of a physical crime scene, the employee's office. During the investigation of this physical crime scene (i.e. **physical investigation**), several digital crime scenes are added to the set of *DCS*, including the laptop of the employee, his smartphone and other digital devices that can be found in his office. The investigator first starts a **digital investigation** on the laptop as it seems to be the digital object with the greatest informational potential. In addition, it is likely that the download was performed on this machine. It is switched off and a copy of its disks is made (**protection** and **documentation**). The investigator then uses the SADFC approach to build the timeline and to analyse it.

The first phase carried out by SADFC is the **specification of events**. It is also the first phase of the event reconstruction process as explained in Section 6.2. The specification of events starts with the **extraction** step which identifies the footprints left by the user's activities. This step is carried out using Plaso on the disk image of the laptop. Listing 6.1 shows the footprints resulting of this step.

```
date, time, timezone, MACB, source, sourcetype, type, user, host, short, desc, version,
filename, inode, notes, format, extra
(E1) 07/07/2015, 15:39:11, UTC, .A., LOG, WinPrefetch, Last Time Executed, -,
WIN-LPAH04, CHROME.EXE was run 122 time(s), Prefetch [CHROME.EXE] was executed -
run count 122 path: \PROGRAM FILES\GOOGLE\CHROME\APPLICATION\CHROME.EXE hash:
0x0548EF22 volume: 1 [serial number: 0xD4420B4A device path:
\DEVICE\HARDDISKVOLUME1], 2, TSK:/Windows/Prefetch/CHROME.EXE-0548EF22.pf, 44136,
-, prefetch, number_of_volumes: 1 volume_device_paths:
[u'\\DEVICE\\HARDDISKVOLUME1'] volume_serial_numbers: [3561098058L] version: 23
prefetch_hash: 88665890
(E2) 07/07/2015, 15:39:20, UTC, .A., WEBHIST, Chrome History, Page Visited, -,
WIN-LPAH04, http://downloadParadise.com/DL/index.html (Download Films for Free.
Wate..., http://downloadParadise.com/DL/index.html (Download Films for Free. Watch
Now!) [count: 46] Host: downloadParadise.com (typed 2 times - not indicating
directly typed count), 2, TSK:/Users/AlanJ/AppData/Local/Google/Chrome/User
Data/Default/History, 43419, -, sqlite, plugin: chrome_history
(E3) 07/07/2015, 15:39:24, UTC, .A., WEBHIST, Chrome History, Page Visited, -,
WIN-LPAH04, http://downloadParadise.com/DL/top10boxOffice.html (Top 10 Box Off...,
http://downloadParadise.com/DL/top10boxOffice.html (Top 10 Box Office) [count: 2]
Host: downloadParadise.com Visit from: http://downloadParadise.com/DL/index.html
(Top 10 Box Office) (URL not typed directly - no typed count), 2,
TSK:/Users/AlanJ/AppData/Local/Google/Chrome/User Data/Default/History, 43419, -,
sqlite, plugin: chrome_history
(E4) 07/07/2015, 15:39:25, UTC, ...B, WEBHIST, Chrome History, File Downloaded, -,
```

```

WIN-LPAH04, C:\Users\AlanJ\Downloads\magnusRex.avi downloaded (681574400 bytes),
http://downloadParadise.com/DL/magnusRex.avi
(C:\Users\AlanJ\Downloads\magnusRex.avi). Received: 681574400 bytes out of:
681574400 bytes., 2, TSK:/Users/AlanJ/AppData/Local/Google/Chrome/User
Data/Default/History, 43419, -, sqlite, plugin: chrome_history
(E5) 07/07/2015, 15:40:33, UTC, .A.., WEBHIST, Chrome History, Page Visited, -,
WIN-LPAH04, http://www.usatoday.com/ (USA TODAY: Latest World
and...), http://www.usatoday.com/ (USA TODAY: Latest World and US News) [count: 0]
Host: usatoday.com (URL not typed directly - no typed count), 2,
TSK:/Users/AlanJ/AppData/Local/Google/Chrome/User Data/Default/History, 43419, -,
sqlite, plugin: chrome_history
(E6) 07/07/2015, 18:01:41, UTC, M..., RECBIN, Recycle Bin, Content Deletion Time, -,
WIN-LPAH04, Deleted file: C:\Users\AlanJ\Downloads\contagioMalware.exe,
C:\Users\AlanJ\Downloads\contagioMalware.exe, 2,
TSK:/$Recycle.Bin/S-1-5-21-1319092878-3599304022-3253320483-1000$IQRSRGM.exe,
43673, -, recycle_bin, file_size: 174054
(E7) 07/07/2015, 18:03:52, UTC, .A.., WEBHIST, Chrome History, Page Visited, -,
WIN-LPAH04, http://www.usatoday.com/story/news/london-bombings (Annivers...),
http://www.usatoday.com/story/news/london-bombings (Anniversary of London 7/7
bombings) [count: 0] Host: usatoday.com (URL not typed directly - no typed count),
2, TSK:/Users/AlanJ/AppData/Local/Google/Chrome/User Data/Default/History, 43419,
-, sqlite, plugin: chrome_history

```

Listing 6.1: Footprints collected in the crime scene using the Plaso toolbox

Then, the **mapping** step is intended to extract relevant information from the footprints and to structure it. In our example, six **objects** can be identified. The first object is a webpage entitled "*Download Films for Free. Watch Now!*" which can be defined as following, $webpage_1 \in O = \{"Download Films for Free. Watch Now!", vl_1\}$ with $vl_1 \in L = \langle http://downloadParaside.com, DL, index.html \rangle$. The second object is a webpage entitled "*Top 10 Box Office*" which can be defined as follows, $webpage_2 \in O = \{"Top 10 Box Office", vl_2\}$ with $vl_2 \in L = \langle http://downloadParaside.com, DL, top10boxoffice.html \rangle$. Then, the download involved two objects which are $webResource_1 \in O = \{681574400, vl_3\}$ with $vl_3 \in L = \langle http://downloadParaside.com, DL, magnusRex.avi \rangle$ and $file_1 \in O = \{681574400, vl_4\}$ with $vl_4 \in L = \langle C:\Users.Andy.Downloads,magnusRex.avi \rangle$ where 681574400 is the size of the objects. The fifth object is a webpage entitled "*USA TODAY: Latest World and US News*" which can be defined as following, $webpage_3 \in O = \{"USATODAY : Latest World and US News", vl_5\}$ with $vl_5 \in L = \langle http://www.usatoday.com/ \rangle$. The sixth object is a webpage entitled "*Anniversary of London 7/7 bombings*" which can be defined as following, $webpage_4 \in O = \{"Anniversary of London 7/7 bombings", vl_6\}$ with $vl_6 \in L = \langle http://www.usatoday.com, story, news, london-bombings \rangle$.

In addition, four **subjects** can be identified. *AlanJ* is a person who can be represented as follows, $person_1 \in S = \{"AlanJ"\}$. This person is the employee who is suspected to download illegally. *Google Chrome*, the *recycle bin* and the *prefetch process* are three processes which have only one attribute, their name such as $process_1 \in S = \{"Google Chrome"\}$, $process_2 \in S = \{"Recycle bin"\}$ and $process_3 \in S = \{"Prefetch"\}$.

Finally, five **events** can be identified. As there is no information about the end date of events, we consider, for the moment, that they are instantaneous and therefore the start date and the end date are equal. First, the launch of Google Chrome can be represented by $event_1 \in E = \{t_{start} = "07/07/2015, 15:39:11", t_{end} = "07/07/2015, 15:39:11", l_e = "WINLPAH04", S_e = \{process_3\}, O_e = \emptyset, E_e = \emptyset\}$. Then four events are defined to represent the visit of the webpages:

- $event_2 \in E = \{t_{start} = "07/07/2015, 15:39:20", t_{end} = "07/07/2015, 15:39:20", l_e = "WINLPAH04", S_e = \{person_1, process_1\}, O_e = \{webpage_1\}, E_e = \emptyset\}$.
- $event_3 \in E = \{t_{start} = "07/07/2015, 15:39:24", t_{end} = "07/07/2015, 15:39:24", l_e = "WINLPAH04", S_e = \{person_1, process_1\}, O_e = \{webpage_2\}, E_e = \emptyset\}$.
- $event_5 \in E = \{t_{start} = "07/07/2015, 15:40:33", t_{end} = "07/07/2015, 15:40:33", l_e = "WINLPAH04", S_e = \{person_1, process_1\}, O_e = \{webpage_3\}, E_e = \emptyset\}$.
- $event_7 \in E = \{t_{start} = "07/07/2015, 18:03:52", t_{end} = "07/07/2015, 18:03:52", l_e = "WINLPAH04", S_e = \{person_1, process_1\}, O_e = \{webpage_4\}, E_e = \emptyset\}$.

The download of the file `magnusRex.avi` can be represented as follows. $event_4 \in E = \{t_{start} = "07/07/2015, 15:39:25", t_{end} = "07/07/2015, 15:39:25", l_e = "WINLPAH04", S_e = \{person_1, process_1\}, O_e = \{webResource_1, file_1\}, E_e = \emptyset\}$. Finally, the deletion of this file can be represented by $event_6 \in E = \{t_{start} = "07/07/2015, 18:01:41", t_{end} = "07/07/2015, 18:01:41", l_e = "WINLPAH04", S_e = \{process_2\}, O_e = \{file_1\}, E_e = \emptyset\}$.

The mapping also identifies relationships between the entities. Figure 6.7 and Figure 6.8 show respectively the relationships between events and subjects and the relationships between events and objects. The **knowledge enhancement** is the last step of the **specification of events**. It aims to deduce new knowledge from the facts identified by the **mapping** step. In this case, several deductions can be made.

For example, the investigator can point out that $event_6$ happens after $event_5$ and before $event_7$. For these two last events, the person that initiated them was identified during the **mapping**, which is not the case for $event_6$. However, considering that $event_6$ is temporally located between $event_5$ and $event_7$, the investigator can presume that $person_1$ (i.e. AlanJ) is also involved in $event_6$. The **specification of events** is now finished and the investigator get, as output, a timeline. This timeline is made of events temporally located in addition to semantic information about entities interacting with events (objects and subjects) and the nature of these interactions.

The next step is the **resizing of the crime scene**. As the employee has downloaded a file from a remote location, this location may be added to the set of the physical crime scenes. The access of this remote location may allow, in particular, the investigator to access to the server logs. However, the investigator chooses not to extend the crime scene and to not consider this remote location for the moment. The investigator decides to investigate completely at first the local crime scene, hoping to find enough information to charge or exculpate the suspect. He considers that the information contained in the local

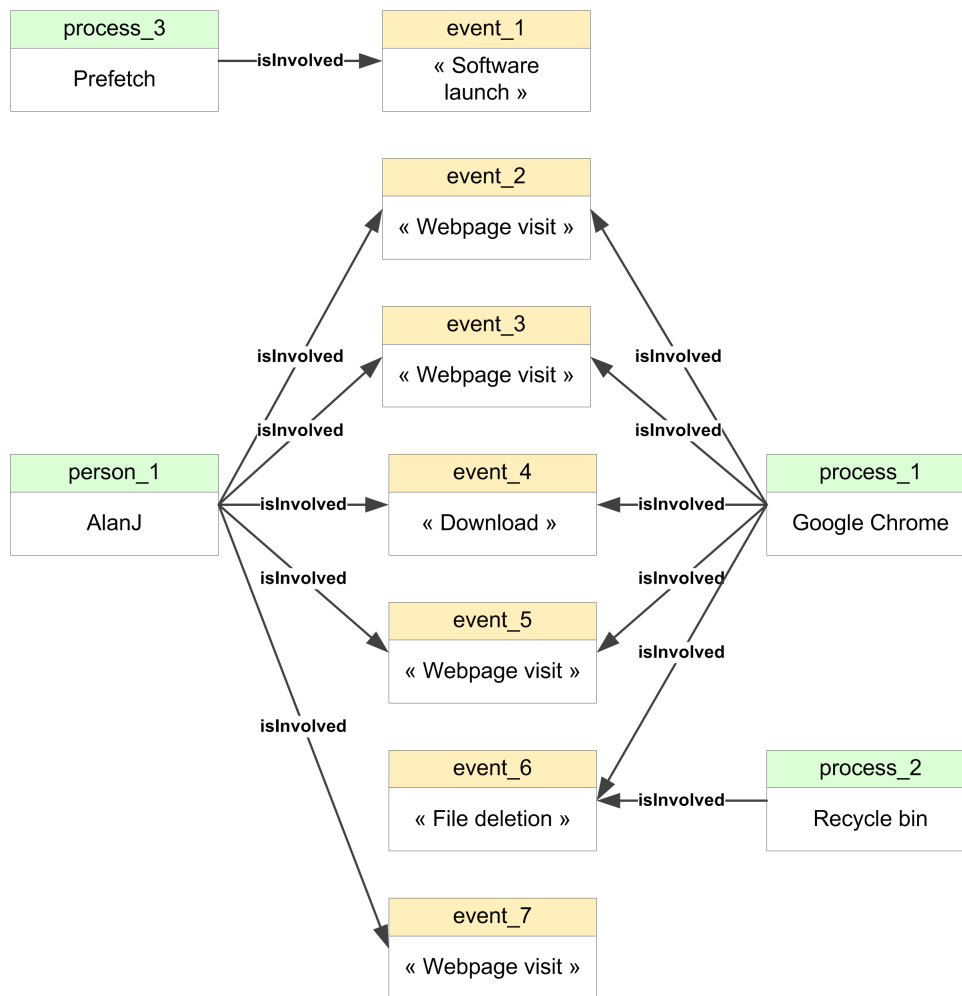


Figure 6.7: Relationships between events and subjects

crime scene is sufficient to answer the main objective of the investigation. In addition, the potential information contained in the remote location has a low value compared to the difficulty to access this new physical crime scene (travel costs, difficulty in obtaining access to the place given its status of private investigator, etc.).

The **timeline analysis** phase then starts. This phase is the second one of the reconstruction of events. The first step of this phase is the **identification of illicit events**. It is carried out manually by the investigator who quickly identifies the event of download among the events composing the timeline. This single event is sufficient to show that a film has been downloaded using the computer. However, this is only a part of the scenario and it is interesting to try to reconstruct the sequence of actions performed by the employee to achieve the download. The purpose of the extraction of the incident scenario is to contextualise the illegal actions in order to understand the circumstances and the modus operandi of the incident. To build the scenario of the incident, the next step is to find **event correlations** to identify events related to the download event. To simplify this case of study, no rules based on expert knowledge are defined. This feature is illustrated

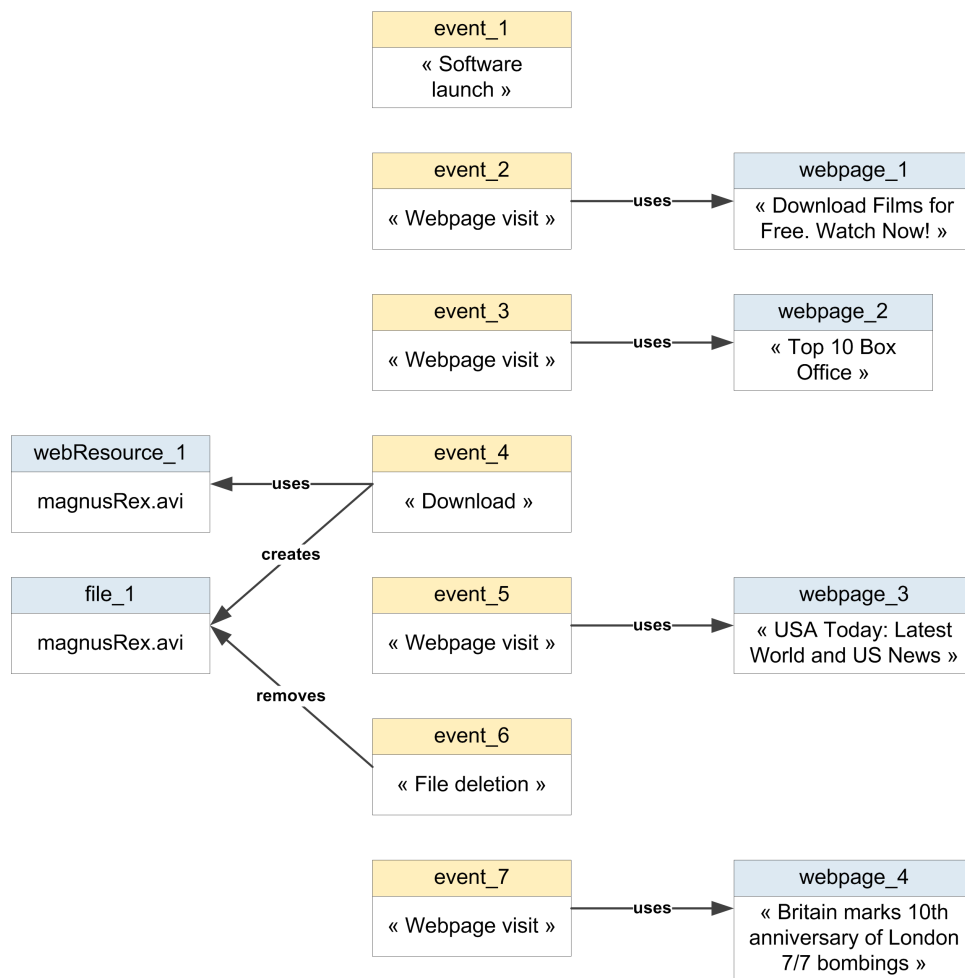


Figure 6.8: Relationships between events and objects

in Chapter 8. Using the tool introduced in Section 6.2.2, five pairs of highly correlated events can be identified (score of correlation > 0.5). Table 6.2 details for each of these pairs the reasons of the high score of correlations.

evt1	evt2	Time	Subjects	Objects
<i>event</i> ₃ (E3)	<i>event</i> ₄ (E4)	Closed in time ($\approx 1s$)	<i>process</i> ₁ and <i>person</i> ₁	<i>webpage</i> ₂ and <i>webResource</i> ₁ are both located in the folder http://downloadParadise.com/DL/
<i>event</i> ₂ (E2)	<i>event</i> ₃ (E3)	Closed in time ($\approx 4s$)	<i>process</i> ₁ and <i>person</i> ₁	<i>webpage</i> ₁ and <i>webpage</i> ₂ are both located in the folder http://downloadParadise.com/DL/
<i>event</i> ₂ (E2)	<i>event</i> ₄ (E4)	Closed in time ($\approx 5s$)	<i>process</i> ₁ and <i>person</i> ₁	<i>webpage</i> ₁ and <i>webResource</i> ₁ are both located in the folder http://downloadParadise.com/DL/
<i>event</i> ₅ (E5)	<i>event</i> ₇ (E7)	Not closed in time ($> 2h$)	<i>process</i> ₁ and <i>person</i> ₁	<i>webpage</i> ₃ and <i>webpage</i> ₄ are both located in the website http://www.usatoday.com/
<i>event</i> ₄ (E4)	<i>event</i> ₆ (E6)	Not closed in time ($> 2h$)	<i>person</i> ₁	<i>file</i> ₁

Table 6.2: Correlation scores for events from Listing 6.1 (score > 0.4)

Two graphs of correlated events appear. The first graph is composed of the download of the file `magnusRex.avi` (*event*₄), the visit of the webpage entitled "*Download Films for Free. Watch Now!*" (*event*₂), the visit of the webpage entitled "*Top 10 Box Office*" (*event*₃) and the deletion of the file named `magnusRex.avi` (*event*₆). The second graph is made of the events representing the visit of the webpage entitled "*USA TODAY: Latest World and US News*" (*event*₅) and the visit of the webpage entitled "*Anniversary of London 7/7 bombings*" (*event*₇).

During the **scenario extraction**, every event which is illicit (belonging to E_{ir}) or correlated to an illicit event (belonging to E_{cr}) is extracted to build the scenario of the incident. As *event*₄ have been identified as an illicit event, the events that are correlated to it are extracted to form the scenario of the incident. In this case, the scenario is formed of events *event*₂, *event*₃, *event*₄ and *event*₆. The **scenario analysis** then allows the investigator to fully understand the scenario of the incident by studying the events composing it:

- He visited two webpages hosted on the website <http://downloadParadise.com>. The URL of these webpages are <http://downloadParadise.com/DL/index.html> and <http://downloadParadise.com/DL/top10boxOffice.html> and the titles of them are respectively "*Download Films for Free. Watch Now!*" and "*Top 10 Box Office*". The first visit took place on 07/07/2015, 15 : 39 : 20 and the second one took place on "07/07/2015, 15 : 39 : 24".
- He then download a file named `magnusRex.avi` from <http://downloadParadise.com/DL/magnusRex.avi> and saved it locally in C:\Users\Andy\Downloads\ local folder. The download was complete as the size received is equal to 681574400 bytes which is the size of the remote resource. It happened on 07/07/2015, 15 : 39 : 25.
- Finally, he later deleted the file `magnusRex.avi` by putting the executable in the recycle bin. It happened on 07/07/2015, 18 : 01 : 41.

Unfortunately, it is not possible to integrate the launch of Google Chrome automatically in the scenario of the incident as this event (**E1**) is not detected as an event correlated to one of the event composing the scenario. However, the use of rules based on expert knowledge, as shown later in this thesis, can address this type of case.

After having extracted and analysed the scenario of the incident, the investigator presents the conclusions of the investigation to the employer during the **post-investigation** phase. The main conclusion is that the employee has downloaded an AVI file with a size of 681574400 bytes using his work machine. This event takes place on 07/07/2015, 15 : 39 : 25. For this, he visited the website <http://downloadParadise.com> which appears to be an illegal downloads site. This website has probably allowed him to get a download link to get the movie. About two hours after the start of the download, the AVI file has been deleted by the user. During these two hours, one can assume that the employee has watched the film or that he has transferred it to an external storage medium. However, the

information is not sufficient to confirm or deny one of these two hypotheses. Moreover, the observed events occurred during a time slot in which the employee is expected to work. This constitutes an additional fault.

6.5/ CONCLUSION

The credibility of the results and their reproducibility are two important criteria to ensure the admissibility of evidence in court. With the aim to build tools that meet these legal requirements inherent to the field of digital forensics, this chapter has introduced the theoretical foundations of SADFC. First, the crime scene and the entities composing it have been defined in order to disambiguate the notions involved in this work. Second, this chapter has proposed a formalisation of the event reconstruction problem. Based on this formalisation, we have then introduced several operators needed to carry out the reconstruction, from the specification of events to their analysis. The introduction of such a formalisation increases the credibility of the results by explaining mathematically how they are produced.

As these operators need to be integrated and interface with the other steps composing the investigation, a new investigation process model, called PaLADIN, has been introduced. This process model is intended to be a framework for the development of automated reconstruction tools. In addition, this model also ensures the reproducibility of evidence by making explicit the steps composing the investigation. It should be noted that, at this stage of the project, the event reconstruction process, as it is defined in PaLADIN cannot be carried out completely by the operators presented in this chapter. Indeed, a step of this process is intended to identify automatically the illicit events composing the timeline. The development of such operator is planned in future work. Therefore, this step of the investigation has to be done manually by the users.

In the next part of the manuscript, the implementation of the event reconstruction process described in PaLADIN, and by extension, the implementation of the proposed reconstruction operators used in it, is presented. This implementation takes the form of an ontology-based architecture composed of several modules, completing tasks from the extraction of footprints to the analysis of the events composing a timeline.

IMPLEMENTATION

ONTOLOGY

This chapter presents the core of the SADFC approach: the ORD2I ontology. As shown in Chapter 5, the use of an ontology in our approach allows to represent accurately the knowledge related to digital events, to structure this knowledge, and to build analysis tools that can both take advantage of generic knowledge and technical knowledge. This ontology is designed to model digital incidents on the one hand and to store information concerning the investigation itself on the other hand (in particular, the reasoning performed to achieve the results of the investigation). The definitions introduced in Chapter 6 are used as a basis for this ontology. It should be noted that this chapter is not intended to give a complete view of ORD2I. The aim of this chapter is to explain the key concepts of it. For more details, the reader can refer to Appendix C where a complete RDF/XML serialisation of the ontology is given. Finally, if the reader is not familiar with the field of semantic web, he can refer to Appendix A where the languages and the technologies used in this chapter are explained in details.

This chapter is structured in the following way. Section 7.1 gives an overview of the ontology by detailing its general structure. This section also details the design choices made during the development of this ontology including the choice of the language Web Ontology Language 2 Rule Language (OWL 2 RL). Section 7.2 introduces the general notions of time and location. Finally, Section 7.3, Section 7.4 and Section 7.5 give a complete view of the concepts and the relations composing the three knowledge layers of ORD2I.

7.1/ OVERVIEW

ORD2I is implemented using the OWL profile OWL 2 RL¹ (a subset of the Web Ontology Language 2 Description Logic (OWL 2 DL)), a language based on *SHROIQ(D)* description logic. The use of this profile is motivated by several reasons, including the availability of sufficient expressiveness to model digital forensics cases. Using OWL 2 RL allows

¹<http://www.w3.org/TR/owl2-profiles/>

to constrain some aspects (class expressions in particular) of the language to ensure decidability and execution time (polynomial) of rule-based reasoning (Motik et al., 2009). The need to operate on large volumes of data and the desire to provide investigators with powerful inference and analysis tools make OWL 2 RL pertinent to implement an ontology for the representation of digital incidents timelines. The OWL2 DL language also allows to give credibility to our model as it has a strong theoretical foundation. Indeed, this ontology language is based on description logics that define formally the semantics of concepts and relations. The components of the ontology and the logic associated to each of them are therefore mathematically defined which allows to check the coherence and the consistency of the knowledge.

ORD2I is built on four bases:

- Design an ontology complete and accurate enough to model any digital incident.
- Integrate generic concepts in the ontology to facilitate analysis (canonical form).
- Take into account specialised knowledge from forensic experts and software developers to integrate the specificities of each information source.
- Model provenance of information.

To meet these needs, ORD2I has a modular design and is divided into three knowledge layers, each allowing to model different kinds of knowledge and answering different goals:

- the Common Knowledge Layer (CKL): this layer integrates generic concepts and described common knowledge about the incident, including the events, the resources used by them and the protagonists involved in them illustrated in Section 7.3. The concepts and the properties composing this layer are based on formal definitions given in Chapter 6.
- the Specialised Knowledge Layer (SKL): this layer handles specialised knowledge and contains technical knowledge about the resources used by the events illustrated in Section 7.4. This layer contains the attributes of the objects as defined in the definition of objects in Chapter 6. SKL is inspired by CybOX (Barnum, 2011), a set of XSD schemas presented in Chapter 3. CybOX integrates detailed information about a wide range of digital objects. This was a source to design the structural elements related to the representation of specialised knowledge about objects.
- the Traceability Knowledge Layer (TKL): this layer aims to answer the need for traceability and credibility by describing precisely every task composing the investigation illustrated in Section 7.5. TKL is inspired from the ontology PROV-O (Lebo et al., 2013), presented in Chapter 3. This ontology was used as a starting point for the design of the structural elements for the representation of information provenance.

The structural level (*TBox*) of these layers is presented below and illustrated throughout this section using G-OWL (Héon and Nkambou, 2013). G-OWL is a graphical language to represent ontology that is only used in this thesis for presentation and explanation purposes. The strengths of this language are to be simpler than serialisations of OWL

such as RDF/XML, Turtle or N3. It is especially designed to model OWL 2 ontologies and is therefore a formal language. The readers can refer to the Appendix B to be introduced to G-OWL. Examples of instantiation (*ABox*) are also given for each layer. For brevity purposes, these examples are illustrated using Turtle² rather than G-OWL. In addition, a serialisation of ORD2I in Turtle is also proposed in the Appendix C.

The namespace prefixes used in ORD2I are given in Table 7.1. These prefixes are used in the rest of the manuscript for readability.

Prefix	Namespace IRI	Description
xsd	http://www.w3.org/2001/XMLSchema#	XML Schema mainly used for datatypes
rdf	http://www.w3.org/1999/02/22-rdf-syntax-ns#	Resource Description Framework
rdfs	http://www.w3.org/2000/01/rdf-schema#	Resource Description Framework Schema
owl	http://www.w3.org/2002/07/owl#	Ontology Web Language
time	http://www.w3.org/2006/time#	OWL-Time ontology
ord2i	http://checksem.u-bourgogne.fr/ord2i#	ORD2I Ontology

Table 7.1: Namespace prefixes used in ORD2I

7.2/ TIME AND LOCATION

The representation of time and location in ORD2I is derived from the definitions given in Section 6.1.2.1 and Section 6.1.2.2 of Chapter 6. Time and location are two very important notions, common to every investigation. Indeed, these allow to locate events, people and objects in time and space. In our ontology, the time is represented using an external ontology named OWL-Time³. OWL-Time is a reference ontology for temporal information modelling and is used by a large number of projects. This ontology is designed specifically for the representation of time information, including the concept of instant, duration, time zones and time unit. For our needs, ORD2I integrates partially OWL-Time. The concepts and properties of OWL-Time used in ORD2I to represent time information about events are illustrated in Figure 7.1.

```

1 @prefix : <http://checksem.u-bourgogne.fr/ord2i#> .
2 @prefix owl: <http://www.w3.org/2002/07/owl#> .
3 @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
4 @prefix xml: <http://www.w3.org/XML/1998/namespace> .
5 @prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
6 @prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
7 @prefix time: <http://www.w3.org/2006/time#> .
8 @prefix ord2i: <http://checksem.u-bourgogne.fr/ord2i#> .
9 @base <http://checksem.u-bourgogne.fr/ord2i> .
10 <http://checksem.u-bourgogne.fr/ord2i> rdf:type owl:Ontology ;
11     owl:imports <http://checksem.u-bourgogne.fr/timeRL> ;
12     owl:versionIRI <http://checksem.u-bourgogne.fr/ord2i/1> .
13 ### http://checksem.u-bourgogne.fr/ord2i#FileDownload50
14 ord2i:FileDownload50 rdf:type ord2i:Event, owl:NamedIndividual ;

```

²<http://www.w3.org/TR/turtle/>

³<http://www.w3.org/TR/owl-time/>

```

15   ord2i:hasEventSubtype "File Download" ;
16   ord2i:hasEventType "Chrome History" ;
17   ord2i:hasDateTime ord2i:Interval52 ;
18   ### http://checksem.u-bourgogne.fr/ord2i#Interval52
19   ord2i:Interval52 rdf:type owl:NamedIndividual, time:Interval ;
20   time:hasBeginning ord2i:Instant51 .
21   time:hasDateTimeDescription ord2i:DateTimeDescription53 ;
22   ### http://checksem.u-bourgogne.fr/ord2i#Instant51
23   ord2i:Instant51 rdf:type owl:NamedIndividual, time:Instant ;
24   time:inXSDDateTime "2015-05-02T08:35:37.000+02:00"^^xsd:dateTime .
25   ### http://checksem.u-bourgogne.fr/ord2i#DateTimeDescription53
26   ord2i:DateTimeDescription53 rdf:type owl:NamedIndividual, time:DateTimeDescription ;
27   time:unitType time:unitSecond .

```

Listing 7.1: Turtle serialisation of temporal information related to the visit of a webpage

Each event composing the incident and each task of the investigation is located in time using the property *ord2i:hasDateTime* linking individuals of classes *ord2i:Event* or *ord2i:InvestigativeOperation* with an interval (represented by *time:Interval*). An interval is defined by a start and an end allowing to represent the duration. *time:Interval* is therefore linked to individuals of *time:Instant* using the properties *time:hasBeginning* and *time:hasEnd*. The classes *time:Interval* and *time:Instant* are both subclasses of the *time:TemporalEntity* class. Each instant is defined by a date and a time represented by the datatype property *time:inXSDDateTime* linking individuals of *time:Instant* with values of the XML Schema datatype *dateTime*. An instant is also described by several information linked to individuals of *time:DateTimeDescription*. This class is used in ORD2I to specify the precision of the date (e.g. second, minute, etc.) using *time:TemporalUnit* and the object property *time:unitType*.

To illustrate the aspects of ORD2I related to time, a Turtle serialisation of temporal information related to the visit of a webpage is given in Listing 7.1. The prefixes given at the beginning of this example are also used in the following examples of this chapter. The event *ord2i:FileDownload50* (lines 14-18) is located in time by the interval *ord2i:Interval52*. This interval has a beginning defined by the instant *ord2i:Instant51* which is equal to "2015-05-02T08:35:37.000+02:00" (Coordinated Universal Time) (lines 20-26). Finally, the description *ord2i:DateTimeDescription53* gives information about the interval (in this case, information about the granularity of it, i.e. "unitSecond") (lines 27-29).

As said in Chapter 6, the location of protagonists and physical or digital objects is an important aspect of the investigation. In our work, we consider two types of locations which are the physical locations and the virtual locations. The elements of ORD2I related to the notion of location are illustrated in Figure 7.2 and Figure 7.3. The ontology contains a hierarchy of five classes allowing to model a location (*ord2i:Location*), a physical location (*ord2i:PhysicalLocation*) or a virtual location (*ord2i:VirtualLocation*). A virtual location can belong to *ord2i:LocalVirtualLocation* (e.g. the path of a local file on a computer) or to *ord2i:RemoteVirtualLocation* (e.g. the URL of a remote resource on a web server).

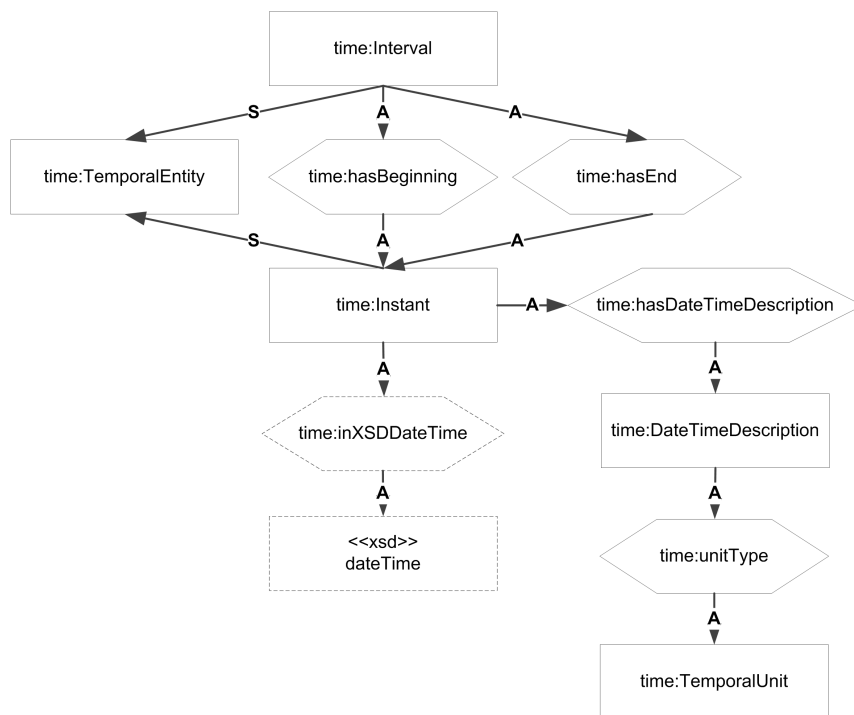


Figure 7.1: Concepts and properties of the OWL-Time ontology used in ORD2I

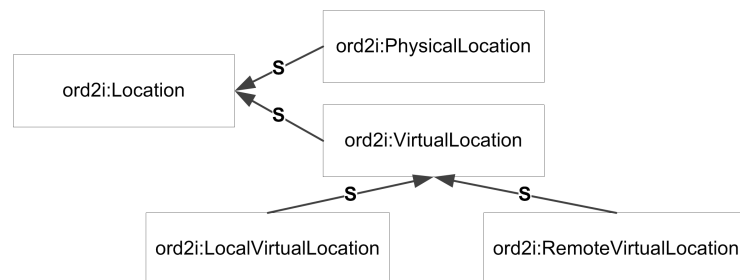


Figure 7.2: Concepts modelling locations in ORD2I

The information related to each location is defined using datatype properties. Regarding virtual locations, every individual of the class *ord2i:VirtualLocation* has a filename and an extension (e.g. ".doc", ".png"). Depending on the type of virtual location, the individual also carries a drive letter and a path on the computer (if it is a local virtual location) or a hostname and an URL (if it is a remote virtual location). Regarding physical locations, each individual of the class *ord2i:PhysicalLocation* has several attributes allowing to locate physically an entity, including a zip code, a country and a city.

To illustrate the use of virtual locations, a Turtle serialisation of location information of the executable file *ord2i:WinExeFile24* (lines 1-3) and the webpage *ord2i:Webpage13* (lines 9-11) are given in Listing 7.2. These two objects are located using respectively the locations *ord2i:Location25* and *ord2i:Location14*. *ord2i:Location25* locates the executable on the file system by giving its filename *chrome.exe* and its path *c:\programfiles\google\chrome\application* (lines 5-8). *ord2i:Location14* locates the

webpage on the remote server by giving its hostname drive.google.com and its url https://drive.google.com/drive/... (lines 13-16).

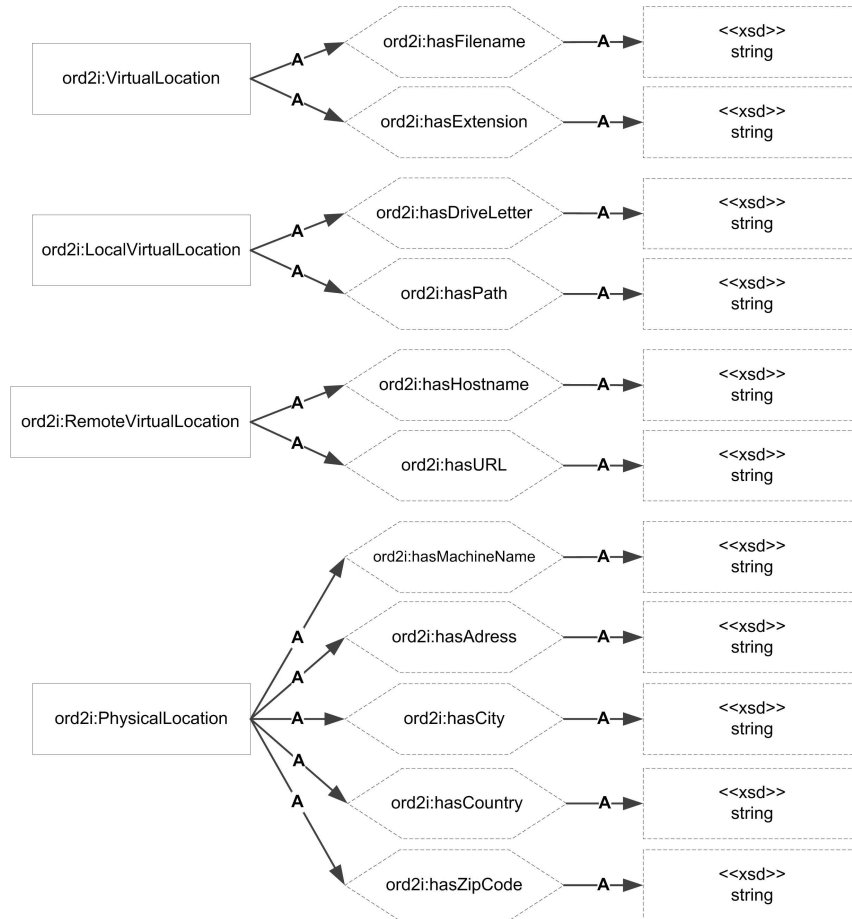


Figure 7.3: Properties modelling locations in ORD2I

```

1  ### http://checksem.u-bourgogne.fr/ord2i#WinExeFile24
2  ord2i:WinExeFile24 rdf:type ord2i:WinExeFile, owl:NamedIndividual ;
3    ord2i:hasLocation ord2i:Location25 .    ...
4  ### http://checksem.u-bourgogne.fr/ord2i#Location25
5  ord2i:Location25 rdf:type ord2i:LocalVirtualLocation, owl:NamedIndividual ;
6    ord2i:hasPath "c:\\program files\\google\\chrome\\application\\" ;
7    ord2i:hasFilename "chrome.exe" .
8  ### http://checksem.u-bourgogne.fr/ord2i#Webpage13
9  ord2i:Webpage13 rdf:type ord2i:Webpage, owl:NamedIndividual ;
10   ord2i:hasLocation ord2i:Location14 .    ...
11  ### http://checksem.u-bourgogne.fr/ord2i#Location14
12  ord2i:Location14 rdf:type ord2i:RemoteVirtualLocation, owl:NamedIndividual ;
13   ord2i:hasURL "https://drive.google.com/drive/%23folders/0b_xlw_i_6lv9itgnhm9hweu" ;
14   ord2i:hasHostname "drive.google.com" .
  
```

Listing 7.2: Turtle serialisation of location information

7.3/ COMMON KNOWLEDGE ABOUT THE INCIDENT

The CKL layer is used to store common knowledge about events that occurred during an incident. This layer contains information on the resources used by them and information on the subjects who participated in each event and time information about events. This layer provides a canonical representation for events using the principle of polymorphism. Due to the variety of sources, all events have a number of specific attributes, but also a set of common characteristics. For example, if we consider an event produced by an Apache server and another event from the Firefox browser, it is difficult to analyse such events to look for new knowledge because each of them has specific characteristics. An Apache event carries out information about IP addresses, which is requested by the connection, whereas Firefox does not need it. However, these two events have also common characteristics (e.g., the time at which the events occurred). All events belong to the same concept which is the general concept *ord2i:Event*. The use of polymorphism creates a consistent representation for all events without removing the specific data of each type of event (specific data is stored in the SKL). This uniform representation of events allows events to be analysed in the same way.

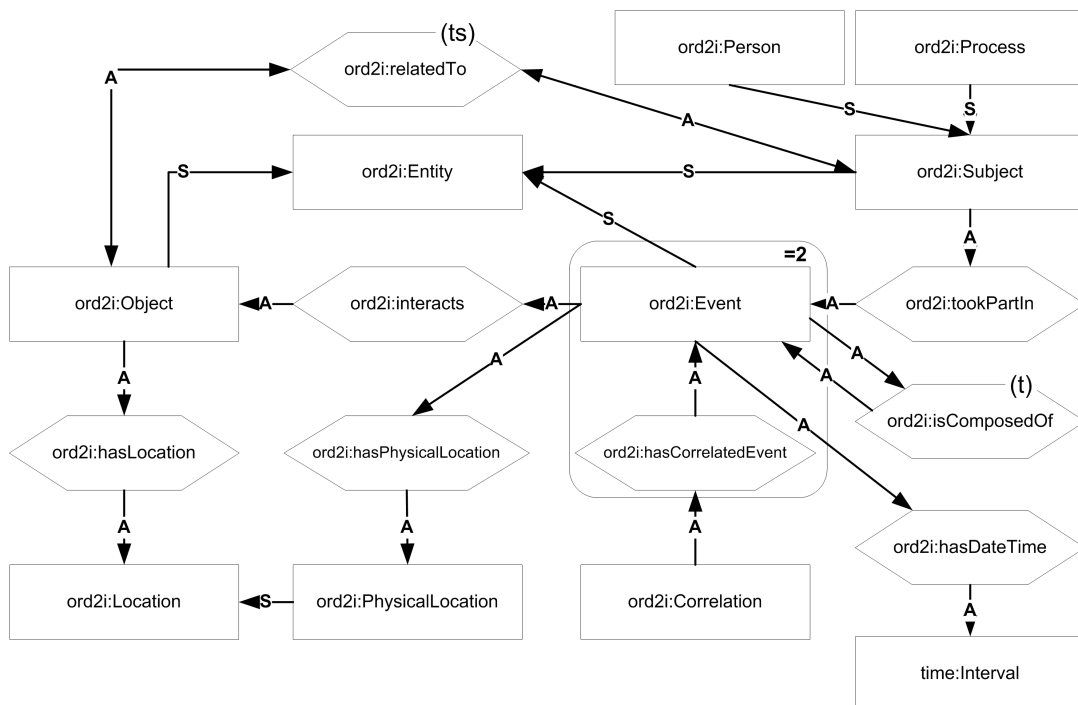


Figure 7.4: Classes and object properties of the CKL

This layer is directly derived from the definitions given in Section 6.1.3 of Chapter 6. The concepts and properties composing CKL are illustrated in Figure 7.4. The Figure 7.5 shows the hierarchy of object properties of CKL and the Figure 7.6 contains the definitions of the datatype properties composing this layer. The notion of *Entity* is a general concept encompassing any entities related to the crime scene. The *ord2i:Entity* class

is therefore subsumed by the *ord2i:Event*, *ord2i:Subject* and *ord2i:Object* classes. First, the *ord2i:Event* class allows to model any digital action happening on a machine. Each event is defined by the type of the action performed given by a type (e.g. "Chrome History", "Recycle Bin", "Link Shortcut", etc.) and a subtype (e.g. "Webpage Visit", "File Download", "File Deletion", "Shortcut Creation", etc.) using the datatype properties *ord2i:hasEventType* and *ord2i:hasEventSubtype*. An event is also linked to the date and the time when the action occurred and the accuracy and the granularity of the date (using concepts and properties from OWL-Time, e.g. *time:Interval*) and a status (e.g. "success", "fail", "error", etc.).

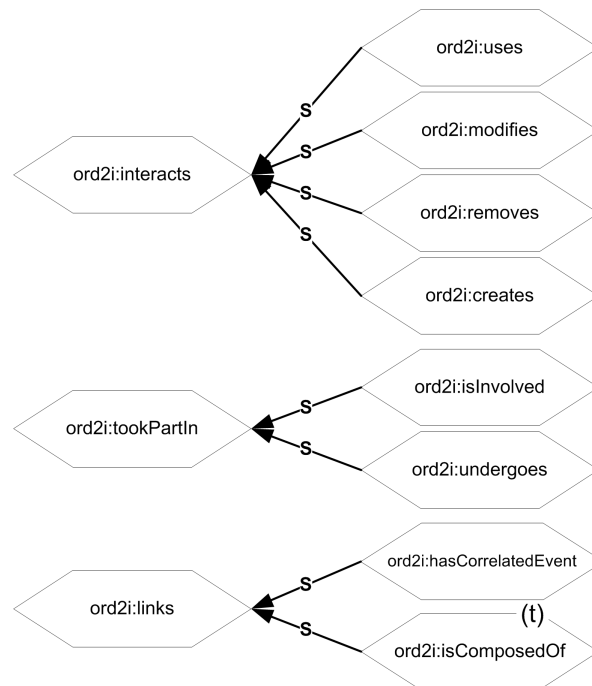


Figure 7.5: Hierarchy of object properties of the CKL

As defined in Chapter 6, events can be linked together by various properties. In our work, we define two different types of semantic properties linking events and subsuming the high-level *ord2i:links* object property. First, two events can be declared as correlated using the *ord2i:Correlation* class. This latter links two correlated events using the *ord2i:hasCorrelatedEvent* object property ("=2" in Figure 7.4 defined the cardinality of this relationship). The notion of correlation is discussed with more details in Section 8.4. Second, an event can be a composition of several events which implies that several constraints are met for these particular events as defined in Chapter 6. The *ord2i:isComposedOf* object property model a composition of events. This property is transitive (small (t) in Figure 7.4) which means that if an event *a* is a part of an event *b* and the event *b* is a part of an event *c*, then *a* is a part of *c*.

The *ord2i:Subject* class is used to model protagonists involved in events. A protagonist can be a person (*ord2i:Person*) or a process (*ord2i:Process*). The characteristics of both

notions are defined using several information, including, for a process, its name and its version, and for a person, his name, his email, his phone etc. The set of datatype properties to define a person or a process is not intended to be exhaustive. The implication of a protagonist in an event is modelled using the *ord2i:tookPartIn* property. This latter is subsumed by two properties allowing to model that a subject participate (*ord2i:isInvolved*) in an event or undergoes (*ord2i:undergoes*) it.

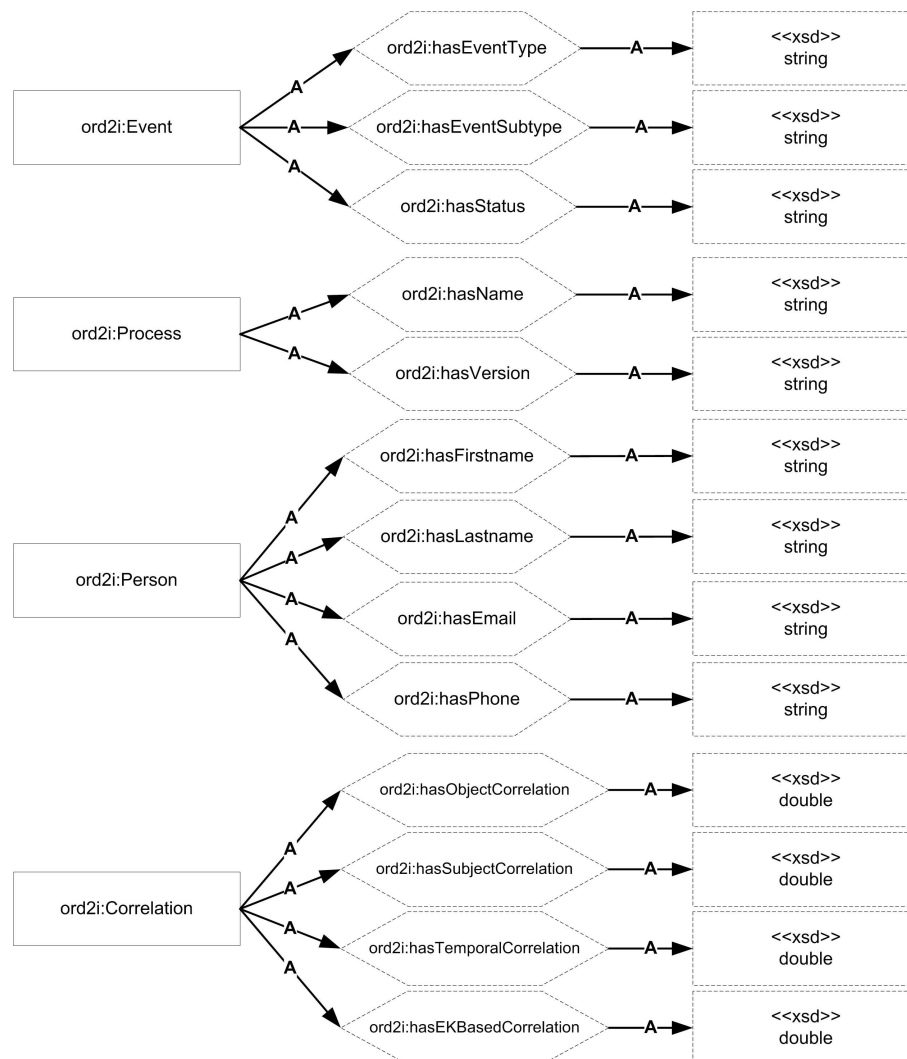


Figure 7.6: Datatype properties of the CKL

The *ord2i:Object* class is used to represent resources that interact (*ord2i:interacts*) with events (e.g. a document edited by Microsoft Word, a bookmark created by Firefox, etc.). An object can be used (*ord2i:uses*), modified (*ord2i:modifies*), removed (*ord2i:removes*) or created (*ord2i:creates*) by an event. The characteristics of the objects are defined in the SKL. Finally, objects and subjects can be directly linked using the property *ord2i:isRelatedTo*. A person can be linked to an object representing his Windows session using this property. It can also be used to link a cookie with the website that generated it or a scheduled task with the related executable. This property is transitive (see the

explanation above about transitivity) and symmetric (small (s) in Figure 7.4) which means that if an entity *a* is related to an entity *b*, *b* is also related to *a*.

To illustrate the use of the CKL, the Listing 7.3 shows the knowledge describing a download. The event *ord2i:FileDownload50* (lines 1-10) carries several information, including its type ("Chrome History") and its subtype (File Download), a description of the event, the date and time of the event (represented by the interval *ord2i:Interval52*) and its physical location (represented by the individual *ord2i:PhysicalLocation6*). Two subjects are involved in *FileDownload50*. The first is an individual *ord2i:Person11* (lines 12-17). This person has a pseudo ("Yoan") and is linked to the Windows user session *ord2i:WinUserAccount12*. Moreover, *ord2i:Person11* is involved in four other events linked using the object property *ord2i:isInvolved*. The second subject is the process *ord2i:Process10* which represents Google Chrome (lines 19-23). This process is involved in four events, including the download. A download is an event manipulating two different objects. This event read the information contained in a remote resource to create locally a copy of it. In this example, *ord2i:FileDownload50* uses (*ord2i:uses*) the remote resource *ord2i:WebResource54* (lines 25-28) to create (*ord2i:creates*) the local file *ord2i:WinExeFile66* (lines 30-33) which is a copy of it.

```

1  ### http://checksem.u-bourgogne.fr/ord2i#FileDownload50
2  ord2i:FileDownload50 rdf:type ord2i:Event, owl:NamedIndividual ;
3    ord2i:hasEventType "Chrome History" ;
4    ord2i:hasEventSubtype "File Download" ;
5    ord2i:hasDescription "File Download(Chrome History) made by Yoan. File
      exploit.exe(c:\\users\\yoan\\downloads\\) downloaded from
      http://www.yoan-chabot.fr/espacepersonnel/exploit.exe" ;
6    ord2i:hasDateTime ord2i:Interval52 ;
7    ord2i:hasPhysicalLocation ord2i:PhysicalLocation6 ;
8    ord2i:uses ord2i:WebResource54 ;
9    ord2i:creates ord2i:WinExeFile66 . ...
10 ### http://checksem.u-bourgogne.fr/ord2i#Person11
11 ord2i:Person11 rdf:type ord2i:Person, owl:NamedIndividual ;
12   ord2i:hasPseudo "Yoan" ;
13   ord2i:isInvolved ord2i:ApplicationPrefetch19, ord2i:FileDownload50 ;
14   ord2i:isInvolved ord2i:WebpageVisit30, ord2i:WebpageVisit42, ord2i:WebpageVisit5 ;
15   ord2i:isRelatedTo ord2i:WinUserAccount12 . ...
16 ### http://checksem.u-bourgogne.fr/ord2i#Process10
17 ord2i:Process10 rdf:type ord2i:Process, owl:NamedIndividual ;
18   ord2i:hasName "Google Chrome" ;
19   ord2i:isInvolved ord2i:FileDownload50 ;
20   ord2i:isInvolved ord2i:WebpageVisit30, ord2i:WebpageVisit42, ord2i:WebpageVisit5 .
21 ### http://checksem.u-bourgogne.fr/ord2i#WebResource54
22 ord2i:WebResource54 rdf:type ord2i:WebResource, owl:NamedIndividual ;
23   ord2i:hasSize "7444016"^^xsd:int ;
24   ord2i:hasLocation ord2i:Location55 . ...
25 ### http://checksem.u-bourgogne.fr/ord2i#WinExeFile66
26 ord2i:WinExeFile66 rdf:type ord2i:WinExeFile, owl:NamedIndividual ;
27   ord2i:hasUsageCounter "1"^^xsd:int ;
28   ord2i:hasLocation ord2i:Location67 . ...

```

Listing 7.3: Knowledge about the download of a file, stored in the CKL

7.4/ SPECIALISED KNOWLEDGE ABOUT THE INCIDENT

The SKL is used to store specialised information on objects used by events. SKL models technical knowledge about every kind of digital objects that can be found in a cyber environment. One advantage of SKL is to reason about events using technical knowledge. For example, IP addresses, file paths or metadata files are all valuable information during analysis. The objective of SKL is to provide the structure needed for their storage. The SKL layer is not intended to be exhaustive and the use of an ontology allows to easily integrate new classes or modify an existing class. The hierarchy of classes included in SKL is illustrated in Figure 7.7 and Figure 7.8. The Figure 7.9 shows the datatype properties composing this layer. This layer provides an important range of classes to represent a large number of digital objects:

- Files (*ord2i:File*) : *ord2i:OLECF*, *ord2i:Link*, *ord2i:ArchiveFile*, *ord2i:ImageFile*, *ord2i:PDFFile*, *ord2i:ExeFile*.
- User account (*ord2i:Account*) : *ord2i:UnixUserAccount*, *ord2i:WinUserAccount*, *ord2i:ComAccount*.
- Objects related to the Web (*ord2i:Web*) : *ord2i:Webpage*, *ord2i:WebResource*, *ord2i:EmailMessage*, *ord2i:Bookmark*, *ord2i:Cookie*, etc.
- Objects related to communications (*ord2i:Communication*) : *ord2i:MMS*, *ord2i:SMS*, *ord2i:Chat*, *ord2i:Call*.
- Registry keys (*ord2i:RegisterKey*).
- etc.

The Listing 7.4 gives knowledge stored in SKL related to three objects: a webpage (*ord2i:Webpage34*) (lines 1-11), a user account (*ord2i:WinUserAccount12*) (lines 12-14) and an executable (*ord2i:WinExeFile66*) (lines 15-23).

The proposed version of the SKL provides a set of datatype properties allowing to describe each of these objects. However, this set is not intended to be complete.

7.5/ REPRODUCIBILITY OF THE INVESTIGATION PROCESS

Finally, the TKL layer stores information about how the investigation is conducted. This includes information about investigative activities, the information used in the investigation and agents involved. For example, this layer is used to memorise how each result of the investigation is produced. The aim of this layer is to satisfy some legal requirements, by ensuring reproducibility of the results by storing all actions taken at each stage of the investigation, and by ensuring that the conclusions have been supported by credible data and evidence. The concepts and object properties composing TKL are illustrated in Figure 7.10. The Figure 7.11 displays the datatype properties associated with the previous

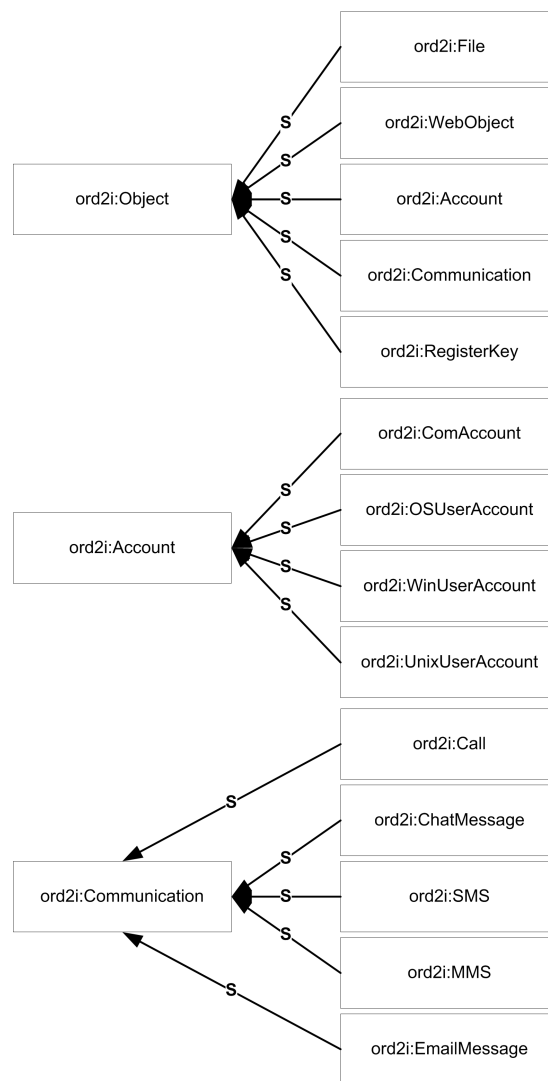


Figure 7.7: Hierarchy of classes of SKL (first part)

classes. The *ord2i:InvestigativeOperation* class allows to represent any task undertaken during a computer forensics investigation (i.e. collect of a footprint, analysis of data using forensic tools, etc.). Each task is characterised by a set of attributes including the type of techniques used (i.e extraction from an information source, inference of new knowledge, event correlation, etc.) via the datatype properties *ord2i:hasTechniqueType* and *hasTechniqueName*, the information source used (i.e. Windows registry, Web browsers histories, etc.) via *ord2i:hasSourceType* and *ord2i:hasSourceName*, the date and the place where the task was performed using the object property *ord2i:hasDateTime*, a description of the task via *ord2i:hasDescription* and a numerical value quantifying the degree of confidence of the result of the task using *ord2i:hasTruthfulness*.

Some tasks performed during an investigation are indeed less reliable than others. It is important to quantify the degree of uncertainty of the results produced by each task so that investigators take precautions when using this new knowledge. For example, in the

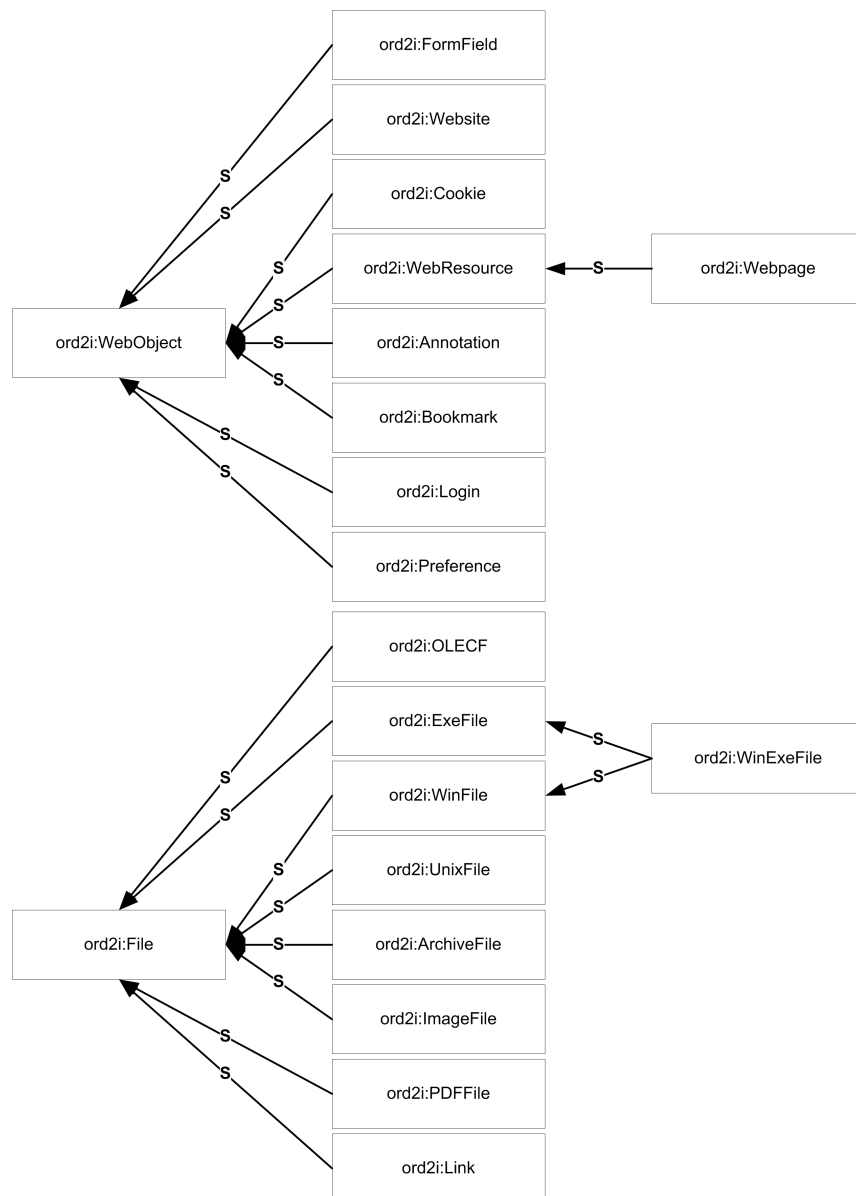


Figure 7.8: Hierarchy of classes of SKL (second part)

case of an operation using as input information that may have been corrupted or obfuscated by attackers, the truthfulness will be low because the results of the operation are unreliable. Each instance belonging to the *ord2i:InvestigativeOperation* class is linked to the tools (*ord2i:Tool*) used via the *ord2i:isPerformedWith* property. Each tool has a name (*ord2i:hasName*) and a version (*ord2i:hasVersion*). The tools are also classified into several types via the *ord2i:hasToolType* (e.g. extraction tool, analysis tool, etc.). The investigators playing a role in the investigation are involved in one or several tasks composing the investigation. The *ord2i:Contribution* class is used to model the contribution of investigators in a given task. Each individual of *ord2i:Contribution* gives information about the role played by the investigator to complete the task via the *ord2i:hasRole* datatype property. Memorizing the contributions made by each person makes, a posteriori and if

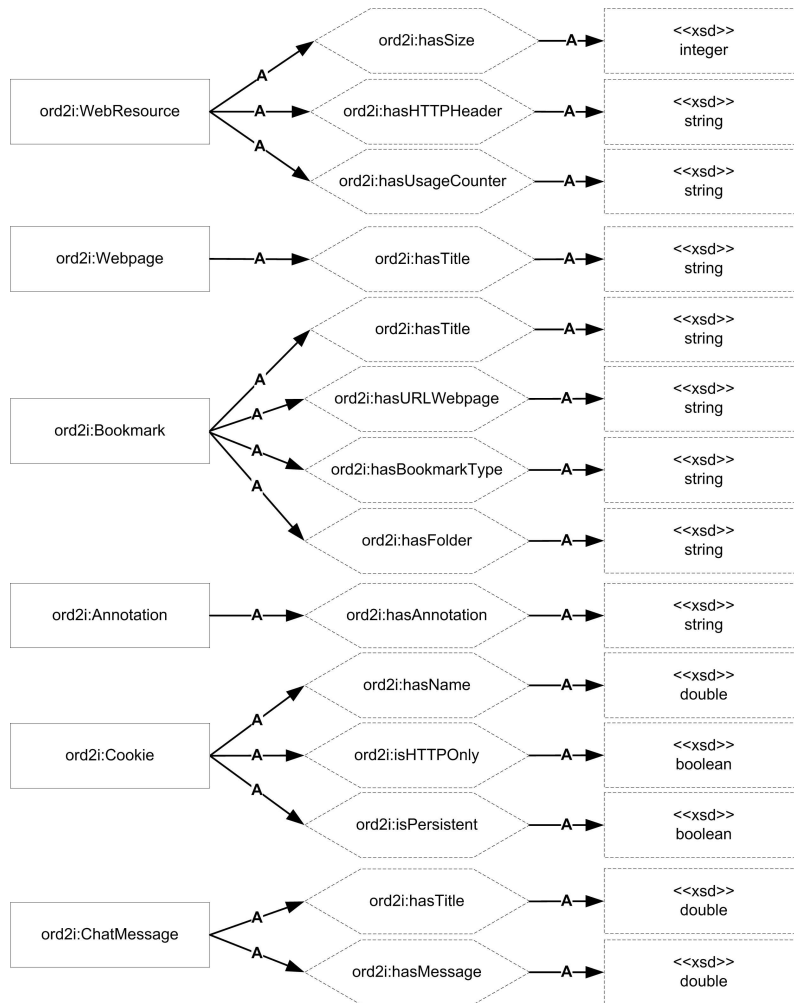


Figure 7.9: Datatype properties of SKL

necessary, to question that person on the results of the contribution.

An investigator is part of a larger organisational scheme. The modelling of the relations between the responders of an investigation is important as it allows to determine who acts on behalf of whom (this is modelled with the *ord2i:actedOnBehalfOf* property. This property is transitive, which means that if an investigator *a* acts on behalf of an investigator *b* and if the investigator *b* acted on behalf of more highly placed investigator, then *a* also acts on behalf of *c*) and for which organisation each responder works (using the *ord2i:hasAffiliation* and *ord2i:Organisation* classes).

Each task belonging to the *ord2i:InvestigativeOperation* class is used to deduce new entities in order to identify the events that happened and the subjects and the objects that have interacted with the events. Therefore, the *ord2i:identifiedBy* object property models the fact that every entity is identified using an investigative operation (e.g. the task of extracting information from a web history can lead to the identification of an event of bookmark creation or the identification of a webpage visited by the user). For some

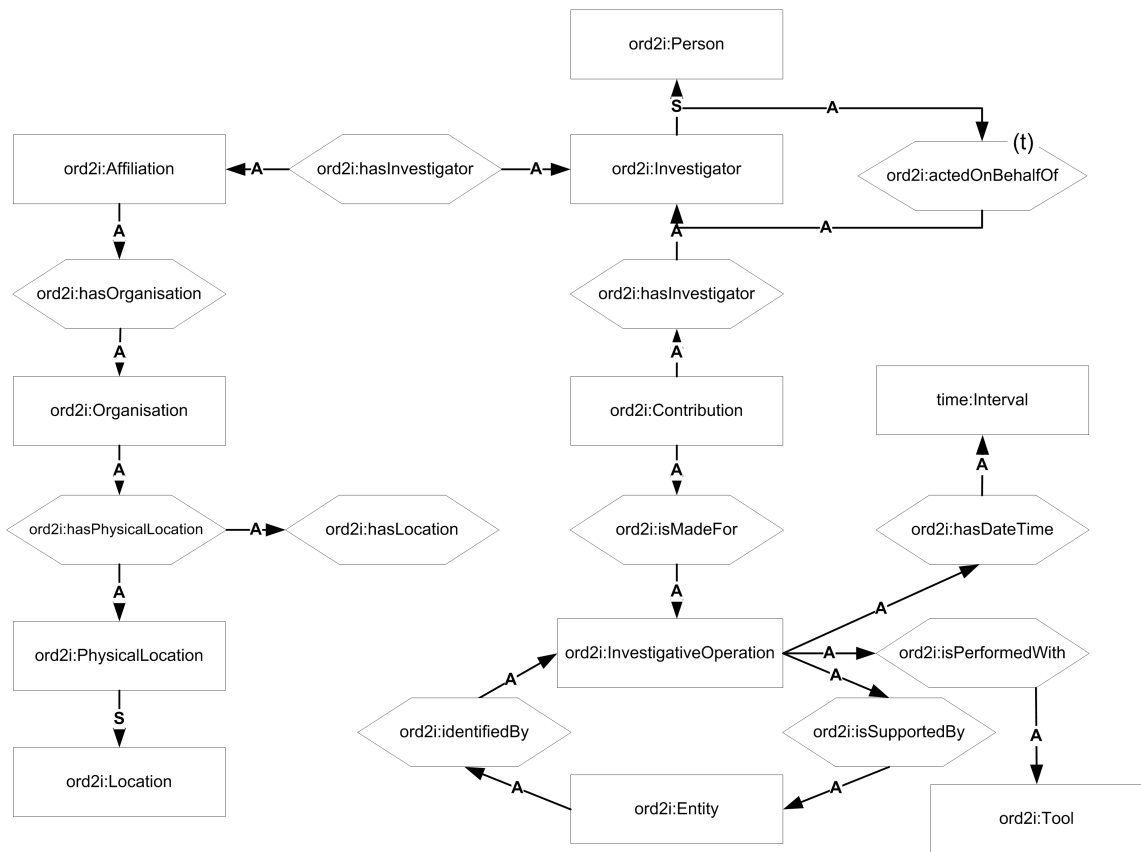


Figure 7.10: Classes and object properties of the TKL

kind of investigative task, investigators have to use information that is already known to deduce new knowledge. The *ord2i:isSupportedBy* object property allows to model this fact by linking an investigative operation to the entities used by it to deduce new knowledge. Thanks to the storing of information about entities used as inputs and outputs of investigative task, it is possible to reconstruct the path of reasoning carried out to produce the final results of the investigation.

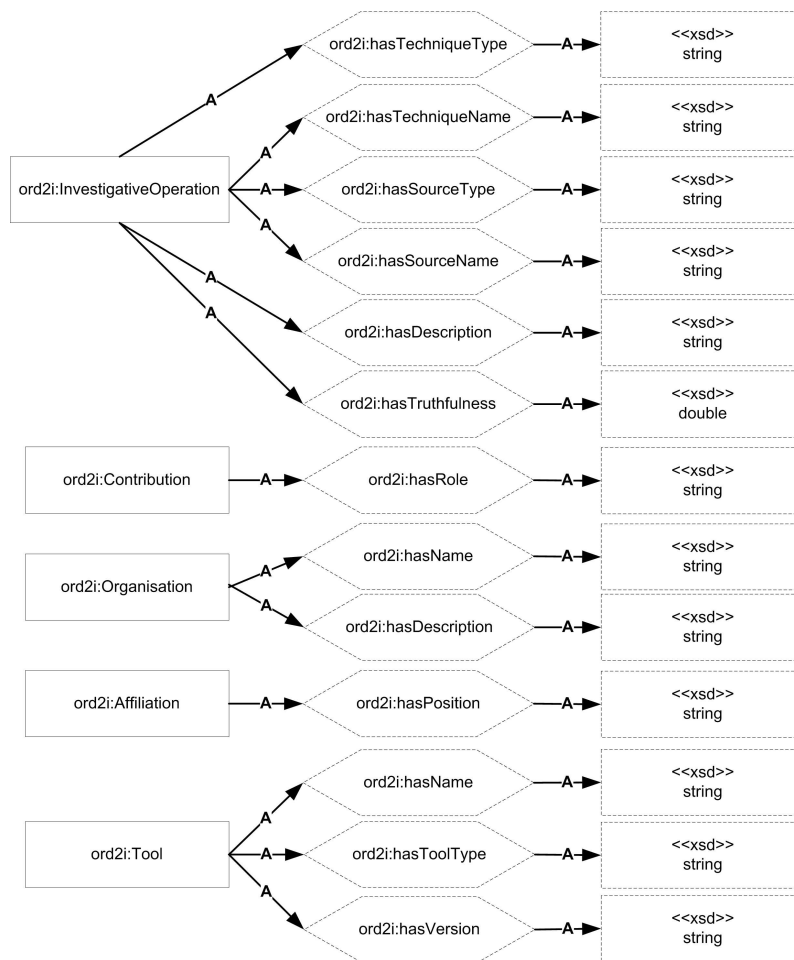


Figure 7.11: Datatype properties of the TKL

```

1  ### http://checksem.u-bourgogne.fr/ord2i#Webpage34
2  ord2i:Webpage34 rdf:type ord2i:Webpage, owl:NamedIndividual ;
3    ord2i:hasUsageCounter "0"^^xsd:int ;
4    ord2i:hasTitle "Yoan Chabot" ;
5    ord2i:hasLocation ord2i:Location35 ;
6    ord2i:isRelatedTo ord2i:Webpage36 . ...
7  ### http://checksem.u-bourgogne.fr/ord2i#Location35
8  ord2i:Location35 rdf:type ord2i:RemoteVirtualLocation, owl:NamedIndividual ;
9    ord2i:hasURL "http://www.yoan-chabot.fr/en/index.php" ;
10   ord2i:hasHostname "www.yoan-chabot.fr" .
11  ### http://checksem.u-bourgogne.fr/ord2i#WinUserAccount12
12  ord2i:WinUserAccount12 rdf:type ord2i:WinUserAccount, owl:NamedIndividual ;
13    ord2i:hasPseudo "Yoan" .
14  ### http://checksem.u-bourgogne.fr/ord2i#WinExeFile66
15  ord2i:WinExeFile66 rdf:type ord2i:WinExeFile, owl:NamedIndividual ;
16    ord2i:hasUsageCounter "1"^^xsd:int ;
17    ord2i:hasLocation ord2i:Location67 . ...
18  ### http://checksem.u-bourgogne.fr/ord2i#Location67
19  ord2i:Location67 rdf:type ord2i:LocalVirtualLocation, owl:NamedIndividual ;
20    ord2i:hasPath "c:\\users\\yoan\\downloads\\" ;
21    ord2i:hasFilename "exploit.exe" .

```

Listing 7.4: Information about the objects stored in the SKL

The Listing 7.5 is an illustration of the use of the TKL. This example shows the ability of this layer to memorise all the stages of the production of knowledge. The information regarding the event *ord2i:ApplicationPrefetch19* (lines 1-11) is first extracted using the Plaso toolbox (*ord2i:Tool0*) (lines 22-25). This extraction is represented by the investigative operation *ord2i:InvestigativeOperation15* (lines 12-21). *ord2i:ApplicationPrefetch19* is linked to *ord2i:InvestigativeOperation15* using the *isIdentifiedBy* property to state that the operation was used to deduce the information regarding this event. As information is directly extracted from the data source by a proven tool, the truthfulness is high. As it will be shown in the next chapter, our tools can make deductions using existing facts. For example, *ord2i:InvestigativeOperation89* (lines 26-34) represents a deduction made with the tool *ord2i:Tool87* (lines 35-38). This operation used the information carried out by two events (the operation is linked to *ord2i:WebpageVisit5* and *ord2i:WebpageVisit30* using the *ord2i:isSupportedBy* object property). The information contained in these events allows to deduce that the user *Person11* (lines 39-45) also plays a role in the event *ord2i:ApplicationPrefetch19* (this tool is presented in Chapter 8).

```

1  ### http://checksem.u-bourgogne.fr/ord2i#ApplicationPrefetch19
2  ord2i:ApplicationPrefetch19 rdf:type ord2i:Event, owl:NamedIndividual ;
3  ord2i:hasEventType "Windows Prefetch" ;
4  ord2i:hasEventSubtype "Application Prefetch" ;
5  ord2i:hasShortDescription "Prefetch, chrome.exe" ;
6  ord2i:hasDescription "Application Prefetch (Windows Prefetch) made by -. Application
   chrome.exe(c:\\program files\\google\\chrome\\application\\) added in Prefetch
   folder" ;
7  ord2i:hasStatus "success" ;
8  ord2i:hasDateTime ord2i:Interval21 ;
9  ord2i:isIdentifiedBy ord2i:InvestigativeOperation15, ord2i:InvestigativeOperation89 ;
10 ord2i:hasPhysicalLocation ord2i:PhysicalLocation6 ;
11 ord2i:uses ord2i:WinExeFile24 .
12 ### http://checksem.u-bourgogne.fr/ord2i#InvestigativeOperation15
13 ord2i:InvestigativeOperation15 rdf:type ord2i:InvestigativeOperation,
   owl:NamedIndividual ;
14 ord2i:hasTruthfulness "100.0"^^xsd:double ;
15 ord2i:hasDescription "Extraction from Windows Prefetch (Windows Prefetch) using
   Plaso" ;
16 ord2i:hasTechniqueType "Information Source" ;
17 ord2i:hasSourceType "Windows Prefetch" ;
18 ord2i:hasSourceName "Windows Prefetch" ;
19 ord2i:hasTechniqueName "Extraction using Plaso" ;
20 ord2i:hasDateTime ord2i:Interval17 ;
21 ord2i:isPerformedWith ord2i:Tool0 .
22 ### http://checksem.u-bourgogne.fr/ord2i#Tool0
23 ord2i:Tool0 rdf:type ord2i:Tool, owl:NamedIndividual ;
24 ord2i:hasName "Plaso" ;
25 ord2i:hasVersion "1.2.0" .
26 ### http://checksem.u-bourgogne.fr/ord2i#InvestigativeOperation89
27 ord2i:InvestigativeOperation89 rdf:type ord2i:InvestigativeOperation,
   owl:NamedIndividual ;
28 ord2i:hasTruthfulness "75.0"^^xsd:double ;
29 ord2i:hasDescription "User Finder using rule n 2 : Let two events A and B, linked
   with a user P, located at both extremities of a chain of events for which the
   user is unknown, then all events composing this chain are linked with P." ;
30 ord2i:hasTechniqueType "Knowledge Enhancement" ;
31 ord2i:hasTechniqueName "User Finder" ;
32 ord2i:hasDateTime ord2i:Interval91 ;
33 ord2i:isPerformedWith ord2i:Tool87 ;
34 ord2i:isSupportedBy ord2i:WebpageVisit30, ord2i:WebpageVisit5 .
35 ### http://checksem.u-bourgogne.fr/ord2i#Tool87
36 ord2i:Tool87 rdf:type ord2i:Tool, owl:NamedIndividual ;
37 ord2i:hasVersion "1.0.0" ;
38 ord2i:hasName "User Finder Tool" .
39 ### http://checksem.u-bourgogne.fr/ord2i#Person11
40 ord2i:Person11 rdf:type ord2i:Person, owl:NamedIndividual ;
41 ord2i:hasPseudo "Yoan" ;
42 ord2i:isInvolved ord2i:ApplicationPrefetch19, ord2i:FileDownload50 ;
43 ord2i:isIdentifiedBy ord2i:InvestigativeOperation1, ord2i:InvestigativeOperation26,
   ord2i:InvestigativeOperation38, ord2i:InvestigativeOperation46 ;
44 ord2i:isInvolved ord2i:WebpageVisit30, ord2i:WebpageVisit42, ord2i:WebpageVisit5 ;
45 ord2i:isRelatedTo ord2i:WinUserAccount12 .

```

Listing 7.5: Traceability of the information handled by the TKL

7.6/ CONCLUSION

This chapter has introduced the ORD2I ontology that is the main component of the SADFC approach. This ontology is designed to represent the events occurring before, during and after a digital incident. One of the specificity of this ontology is its logical separation into three knowledge layers: a layer storing generic information about the entities, a layer storing technical knowledge about digital objects and a layer that stores information on how the investigation is conducted. The proposed ontology meets several needs highlighted in Chapter 3. It structures the information contained in the knowledge base using a schema composed of concepts, properties and logical constraints. It also allows to meet the traceability requirements by storing information about each task conducted during the investigation.

To carry out the reconstruction of events, this ontology must be associated with operators working on the knowledge contained in it. In the next chapter, an architecture, called ANNALIST, is presented. This architecture provides tools to populate the ontology and to perform inference and analysis tasks.

ARCHITECTURE

To carry out the reconstruction of events, the ORD2I ontology needs to be integrated in an ecosystem providing tools to instantiate it and analyse its knowledge. This chapter introduces an architecture called ANNALIST. This architecture is the implementation part of the SADFC approach. It aims to answer the needs highlighted in Chapter 3. To reach this objective, it takes advantage of the inherent qualities of the ontology to provide advanced analysis tools and intuitive visualisation tools to users. The tools provided in ANNALIST are based on theoretical elements presented in Chapter 6.

This chapter is structured as follows. Section 8.1 gives an overview of ANNALIST by introducing the four layers composing it. These layers are then presented in details in dedicated sections. Each of them provides functions carrying out a part of the reconstruction process. Section 8.2 first explains how the extraction of footprints is made using the Plaso toolbox and ad-hoc bridges. Section 8.3 then illustrates the instantiation of the ontology using the extracted information. In particular, it shows how each layer of the ontology is populated to ensure the completeness of the description of the incident in addition to the traceability of the information. Section 8.4 presents how the knowledge contained in the knowledge base is enhanced and analysed. Section 8.5 finally describes the tools provided to manipulate and visualise the knowledge of the base and the conclusions produced by the analysis tools.

It is important to note that the objective of this chapter is not to give technical details about the implementation of the architecture. This chapter only aims to illustrate the operation of the tools composing ANNALIST and the interactions with the knowledge base.

8.1/ OVERVIEW

The proposed architecture, illustrated in Figure 8.2, is capable to automatise the reconstruction of events and to assist the investigators throughout the analysis and the interpretation of the produced timeline. To reach this objective, ANNALIST takes advantage of the ORD2I ontology detailed in Chapter 7. This architecture is made of four layers

(boxes in light grey) and fourteen modules. The modules represented by white boxes and solid black lines are operational modules. The modules represented by white boxes and dashed black lines are modules not yet implemented. The data streams are represented using solid black arrows. ANNALIST implements in its layers the reconstruction steps of the process model PaLADIN (and by extension, the formal operators defined in Chapter 6 to carry out the reconstruction). The relationships between the different modules of this architecture and the steps of the reconstruction process defined in PaLADIN are illustrated in Figure 8.1.

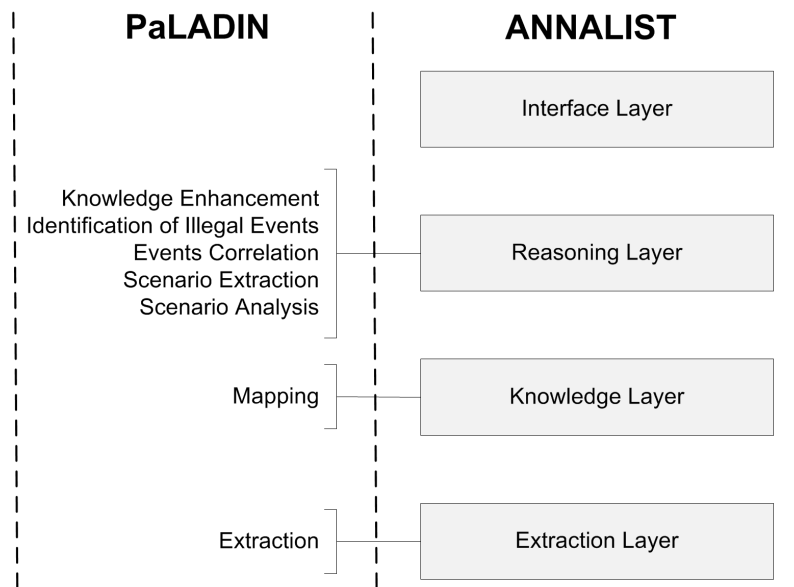


Figure 8.1: Connexion between PaLADIN and ANNALIST

Each layer composing ANNALIST is discussed in the following sections. The first component of ANNALIST is the **extraction layer** which extracts information from footprints contained in heterogeneous source in an automated way. Thanks to Plaso, this layer can handle a large number of information sources, including web browser histories, windows events logs, Skype conversation histories and others. This layer is presented in Section 8.2. The second component is the **knowledge layer** used to instantiate the ontology from information extracted by the extraction layer. This layer also managed aspects regarding traceability of the information in order to ensure the reproducibility of the final results. This layer is presented in Section 8.3. After the instantiation of the ontology, we obtain a knowledge graph representing the information collected from the crime scene. Using the properties of this graph, the **reasoning layer** provides tools to make deductions from the existing knowledge and to draw conclusions. The reasoning capabilities of ANNALIST are discussed in Section 8.4. The last component of the architecture is the **interface layer** which provides visualisation and browsing facilities to investigators to allow them to manipulate the data in an intuitive and efficient way. The aspects regarding the interface are discussed in Section 8.5.

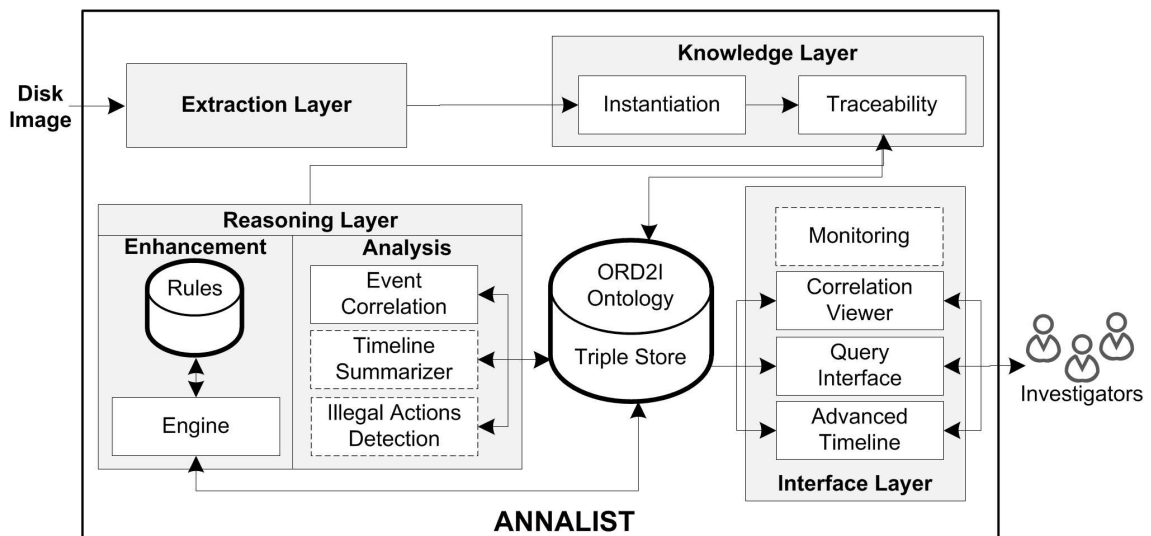


Figure 8.2: Overview of ANNALIST

8.2/ EXTRACTION

The integration of automated extraction techniques is critical to process large volumes of data extracted during an investigation. The extraction phase used in the SADFC approach is a sequential process, illustrated in Figure 8.3 (the grey boxes are modules from the toolbox Plaso), starting with the collection of digital footprints found on a machine and finishing by the serialisation of the relevant information in an appropriate format.

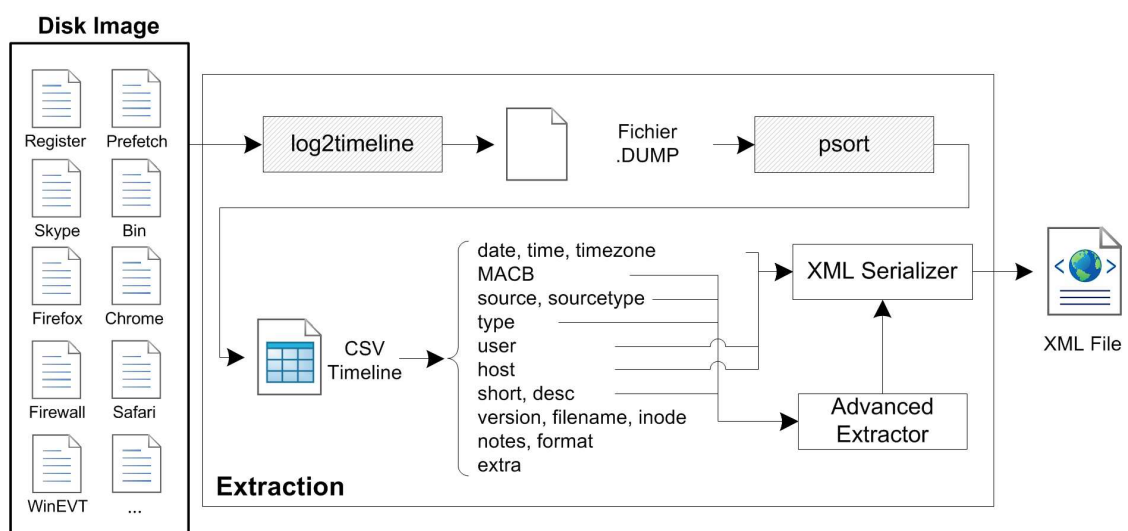


Figure 8.3: Extraction layer

8.2.1/ INFORMATION SOURCES

In a digital crime scene, the information is spread across various sources. From a disk image under investigation, it is possible to extract gigabytes of data from a large number of sources including software logs, file systems, etc. Some sources are more difficult to deal with than others. Indeed, structured sources such as databases or XML files are easy to handle with appropriate parsers. On the other hand, sources such as instant messaging histories or social networks require complex algorithms involving natural language processing, pictures require image processing algorithms to access and understand the content of them. A non-exhaustive description of relevant information that can be extracted from digital systems is given below.

First, the **activities** of a user on a system can be studied using several sources. Operating systems record a lot of information about events occurring on a machine. From a machine using the Windows operating system for example, it is possible to get information about user's and software's activities using event logs (which record information about various kinds of events such as session logging, start/stop service or software, error occurred during the execution of a program, installation of a new software, etc.), system and software configuration using the registry and software launched recently using the prefetch folder. In complement to OS footprints, information about the user's activities are also available in logs of software which are rich source of footprints. For example, antivirus logs contain information about exploits and malicious software detected on the computer. Apache or Microsoft IIS server logs can be used to obtain information about the queries sent to a server. For their part, the files found in a crime scene provide answers to a wide range of questions through the study of their content and their metadata (which allow to know how, when and by whom a file was created, used or modified).

Second, the **behaviour of a user and his interests** can be studied using information contained in files or databases used by browsers to work. Browsers footprints can be used to know the user's interests by studying the user's browsing history (websites visited, date of each visit, etc.), bookmarks and forms filled by the user (e.g. search field, registration form, etc.). Regarding the contents of fields, however, the highly dependent semantics of data make its usage difficult (for example, data entered into the field of a search engine gives information about the user's interests, while a field of a registration form (e.g., to create a website account) may give private information about the user). Links between illegal applications and the remote site that provide these applications can be identified thanks to information about download activities from browser. Browsers footprints also allow to quantify the importance of a webpage for a user. For this purpose, investigators can study bookmarks and user's browsing preferences (zoom used for navigation, character encoding, etc.) to determine which websites are important for the user. A website for which preferences are assigned can be considered as a significant website for the user. The preferences allow to dissociate the accidental visits (e.g. the user has

clicked on a link by accident) from intentional visits (this information may be valuable to determine if the suspect is responsible or not). The footprints left by logging can also be used for this purpose. Indeed, information about all connection pages for which the user has requested to retain his user name and password are registered by browsers. Identifiers can be valuable information if successful decryption techniques are used.

To manage all these sources and take advantage of their information to achieve the objectives of the investigation, the tool *log2timeline* (Gudhjonsson, 2010), proposed as a component of Plaso, is used. *log2timeline* collects information from many sources including: sources inherent to the operating system (e.g. registry, file system, recycle bin, etc.); histories, cookies and cache files of web browsers; files and logs generated by various software including Skype, Google Drive, etc. It should be noted that Plaso can not handle itself all sources of information. However, it remains the tool able to handle the largest number of sources. The architecture we propose handles a part of all the sources handled by Plaso, in order to reduce the time of development of the prototypes. In Appendix D, the list of information sources handled by Plaso and ANNALIST are given. This appendix also presents in detail each information source managed by ANNALIST. In particular, it shows the information that is extracted from each source and the relevance of each information to reach the objectives of the investigation.

8.2.2/ INFORMATION EXTRACTION OF DIGITAL FOOTPRINTS USING THE PLASO TOOLBOX

In order to handle a wide range of information sources, *log2timeline* implements a large set of dedicated extractors. The use of multiple and dedicated parsers allows to take into account specificity of each source while allowing to handle heterogeneous sources. The result is a DUMP file containing all the footprints retrieved from the disk image. Then, transforming the result is necessary to make the data usable by downstream processes. For this, the tool *psort* of Plaso is used. This tool allows to serialise the data produced by *log2timeline* in many formats including CSV. An example of the output of the tool is given in Listing 8.1.

```
date, time, timezone, MACB, source, sourcetype, type, user, host, short, desc, version,
filename, inode, notes, format, extra
(E1) 03/04/2015, 08:13:02, UTC, M..., FILE, NTFS_DETECT mtime;ctime, mtime;ctime, -,
WIN-LPAH04KASIA, /Windows/System32/wbem/Repository/OBJECTS.DATA,
TSK:/Windows/System32/wbem/Repository/OBJECTS.DATA, 2,
TSK:/Windows/System32/wbem/Repository/OBJECTS.DATA, 42621, -, filestat, allocated:
True fs_type: NTFS_DETECT size: [14909440L]
(E2) 03/04/2015, 08:13:13, UTC, .A..., LOG, WinPrefetch, Last Time Executed, -,
WIN-LPAH04KASIA, CHROME.EXE was run 122 time(s), Prefetch [CHROME.EXE] was executed
- run count 122 path: \PROGRAM FILES\GOOGLE\CHROME\APPLICATION\CHROME.EXE hash:
0x0548EF22 volume: 1 [serial number: 0xD4420B4A device path:
\DEVICE\HARDDISKVOLUME1], 2, TSK:/Windows/Prefetch/CHROME.EXE-0548EF22.pf, 44136,
-, prefetch, number_of_volumes: 1 volume_device_paths:
[u'\\DEVICE\\HARDDISKVOLUME1'] volume_serial_numbers: [3561098058L] version: 23
```

- prefetch_hash: 88665890
- (E3) 03/04/2015, 08:13:14, UTC, .A., WEBHIST, Chrome Cookies, Last Access Time, -, WIN-LPAH04KASIA, effectivenessmeasure.net (t), http://effectivemeasure.net/ (t) Flags: [HTTP only] = False [Persistent] = True, 2, TSK:/Users/UserX/AppData/Local/Google/Chrome/User Data/Default/Cookies, 43546, -, sqlite, secure: False host: effectivenessmeasure.net path: / plugin: chrome_cookies
- (E4) 03/04/2015, 08:13:16, UTC, ...B, FILE, NTFS_DETECT crtime;atime, crtime;atime, -, WIN-LPAH04KASIA, /Users/UserX/AppData/Local/Google/Chrome/User Data/Default/Cache/f.0000e4, TSK:/Users/UserX/AppData/Local/Google/Chrome/User Data/Default/Cache/f.0000e4, 2, TSK:/Users/UserX/AppData/Local/Google/Chrome/User Data/Default/Cache/f.0000e4, 16199, -, filestat, allocated: True fs_type: NTFS_DETECT size: [122949L]
- (E5) 03/04/2015, 08:13:24, UTC, .A., WEBHIST, Chrome History, Page Visited, -, WIN-LPAH04KASIA, http://malwareWebsite.com/MalwareDL/index.html (Download Malware Sources and Malw..., http://malwareWebsite.com/MalwareDL/index.html (Download Malware Sources and Malware Binaries) [count: 2] Host: malwareWebsite.com (typed 2 times - not indicating directly typed count), 2, TSK:/Users/UserX/AppData/Local/Google/Chrome/User Data/Default/History, 43419, -, sqlite, plugin: chrome_history
- (E6) 03/04/2015, 08:13:27, UTC, .A., WEBHIST, Chrome History, Page Visited, -, WIN-LPAH04KASIA, http://malwareWebsite.com/MalwareDL/contagio.html (Download Malware Sources and M..., http://malwareWebsite.com/MalwareDL/contagio.html (Download Malware Sources and Malware Binaries) [count: 0] Host: malwareWebsite.com Visit from: http://malwareWebsite.com/MalwareDL/index.html (Download Malware Sources and Malware Binaries) (URL not typed directly - no typed count), 2, TSK:/Users/UserX/AppData/Local/Google/Chrome/User Data/Default/History, 43419, -, sqlite, plugin: chrome_history
- (E7) 03/04/2015, 08:13:28, UTC, ...B, WEBHIST, Chrome History, File Downloaded, -, WIN-LPAH04KASIA, C:\Users\UserX\Downloads\contagioMalware.exe downloaded (174054 bytes), http://malwareWebsite.com/MalwareDL/contagioMalware.exe (C:\Users\UserX\Downloads\contagioMalware.exe). Received: 174054 bytes out of: 174054 bytes., 2, TSK:/Users/UserX/AppData/Local/Google/Chrome/User Data/Default/History, 43419, -, sqlite, plugin: chrome_history
- (E8) 03/04/2015, 08:14:12, UTC, .A., LOG, WinPrefetch, Last Time Executed, -, WIN-LPAH04KASIA, CONTAGIOMALWARE.EXE was run 1 time(s), Prefetch [CONTAGIOMALWARE.EXE] was executed - run count 1 path: \USERS\USERX\DOWNLOADS\CONTAGIOMALWARE.EXE hash: 0x82B5008B volume: 1 [serial number: 0xD4420B4A device path: \DEVICE\HARDDISKVOLUME1], 2, TSK:/Windows/Prefetch/CONTAGIOMALWARE.EXE.pf, 51001, -, prefetch, number_of_volumes: 1 volume_device_paths: [u'\\DEVICE\\HARDDISKVOLUME1'] volume_serial_numbers: [3561098058L] version: 23 prefetch_hash: 2192900235
- (E9) 03/04/2015, 08:14:37, UTC, .A., WEBHIST, Chrome History, Page Visited, -, WIN-LPAH04KASIA, http://www.bbc.co.uk/news/ (BBC News - Home), http://www.bbc.co.uk/news/ (BBC News - Home) [count: 0] Host: www.bbc.co.uk (URL not typed directly - no typed count), 2, TSK:/Users/UserX/AppData/Local/Google/Chrome/User Data/Default/History, 43419, -, sqlite, plugin: chrome_history
- (E10) 03/04/2015, 08:14:47, UTC, M..., RECBIN, Recycle Bin, Content Deletion Time, -, WIN-LPAH04KASIA, Deleted file: C:\Users\UserX\Downloads\contagioMalware.exe, C:\Users\UserX\Downloads\contagioMalware.exe, 2, TSK:/\$Recycle.Bin/S-1-5-21-1319092878-3599304022-3253320483-1000/\$IQRSRGM.exe, 43673, -, recycle_bin, file_size: 174054
- (E11) 03/04/2015, 08:15:29, UTC, ...B, FILE, NTFS_DETECT crtime;mtime;ctime;atime, crtime;mtime;ctime;atime, -, WIN-LPAH04KASIA, /\$Recycle.Bin/S-1-5-21-1319092878-3599304022-3253320483-1000/\$I2R3YRT.exe, TSK:/\$Recycle.Bin/S-1-5-21-1319092878-3599304022-3253320483-1000/\$I2R3YRT.exe, 2, TSK:/\$Recycle.Bin/S-1-5-21-1319092878-3599304022-3253320483-1000/\$I2R3YRT.exe, 43602, -, filestat, allocated: True fs_type: NTFS_DETECT size: [544L]

```
(E12) 03/04/2015, 08:15:51, UTC, .A.., FILE, NTFS.DETECT atime, atime, -,
      WIN-LPAH04KASIA, /Users/UserX/Documents/Meeting report/report february.txt,
      TSK:/Users/UserX/Documents/Meeting report/report february.txt, 2,
      TSK:/Users/UserX/Documents/Meeting report/report february.txt, 43612, -, filestat,
      allocated: True fs_type: NTFS.DETECT size: [32L]
```

Listing 8.1: Data produced by *log2timeline* and formatted using *psort*

This dataset was constructed from a disk image of a virtual machine. Each entry of this file describes an action that has occurred:

- **E1, E4, E11 and E12** : Manipulation of files by the file system.
- **E2** : Execution of the process *chrome.exe* corresponding to the start of the web browser *Google Chrome*.
- **E3** : Access to a cookie by the process *chrome.exe*.
- **E5 and E6** : Visit of two webpages hosted on the website <http://malwareWebsite.com>.
- **E7** : Download of the executable *contagioMalware.exe* from <http://malwareWebsite.com/MalwareDL/contagioMalware.exe> locally saved on the *Download* local folder.
- **E8** : Execution of the process *contagiomalware.exe* corresponding to the use of the malware previously downloaded.
- **E9** : Visit of the homepage of the website <http://www.bbc.co.uk>.
- **E10** : Deletion of the file *contagioMalware.exe* previously downloaded and launched.

8.2.3/ ADVANCED INFORMATION EXTRACTION AND SERIALISATION

The CSV serialisation is structured in seventeen attributes:

- *date, time* and *time zone*: temporal information.
- *MACB*: MAC information. This field provides information on the event's purpose. MACB is derived from MAC notation used in file systems and means Modified, Accessed, Changed and Birth (Creation time). Note that all file systems do not provide the date of creation.
- *source* and *sourcetype*: description of the source of the information.
- *type*: short description of the action performed by the event (e.g. "MSIE Cache File URL record", "Last Time Executed", etc.).
- *user*: the user who initiated the action.
- *host*: name of the host machine.
- *short* and *desc*: description of the event.
- *version, filename, inode, notes* and *format*: information about the tools and the files used during the extraction.
- *extra*: extra information about the entry.

Some fields (*date*, *time*, etc.) do not require additional processing as they are highly structured. Other fields such as the *desc* field require more treatments because their content depends on the type and the status of the event. The variety of their content makes the extraction of the knowledge difficult. A second issue with the timeline produced by Plaso is its size. Indeed, a timeline produced using these tools from a disk image of a machine that operated about thirty minutes contained about 300,000 entries. The analysis of the produced file is almost done manually (e.g. *grep*, search by dates, ad hoc analysis tools such as analysis of browser search history, etc.) by investigators and the interpretation of the timeline is therefore tedious and complex.

With the aim at facilitating the instantiation of ORD2I from the data produced by Plaso, we have developed and integrate into our process an intermediate step of information extraction. For this, the structured information such as temporal information (*date*, *time* and *timezone*), information about the user (*user*), information about the host (*host*) and information about the type of the event (*source* and *sourcetype*) are first extracted. The information contained in less structured fields, such as *desc* is then extracted. As the content of the *desc* field depends on the source of information and the type of the event, a set of regular expression is defined to properly extract the knowledge. As an example, we give the regular expressions and the information extracted during the decomposition of the *desc* of the twelve events presented in Listing 8.1.

Regarding the entires **E1**, **E4**, **E11** and **E12** representing the manipulation of a file by the file system, the extraction pattern 1 is used to extract from *desc* the absolute path of the file concerned with the operation.

Pattern 1. *(TSK:[{filePath}],)**

For the entries **E2** and **E8** representing both the prefetch of an executable file, the extraction pattern 2 is used to extract the name, the path and the hash of the executable, the number of execution and information about the volume containing the executable.

Pattern 2. *Prefetch [{executableName}] was executed - run count {runCount} path: {path} hash: 0x{prefetchHash} {infosVolumes}*

The pattern 3 is used to extract from the *desc* field of **E3** the name of the cookie and the url of the webpage related to it and two flags.

Pattern 3. *{url} ({cookieName}) Flags: [HTTP only] = {httpOnly} [Persistent] = {persistent}*

Regarding the entries **E5**, **E6** and **E9** representing the visit of a webpage using Google Chrome, the extraction pattern 4 is used to extract the URL and the title of the webpage, a visit counter, the hostname of the webpage, the webpage visited previously (in addition to its hostname) and extra information indicating if the URL of the webpage was typed directly or not in the search bar of the web browser.

Pattern 4. `{url} ({title}) [count: {visitCounter}] Host:{host} Visit from: {fromVisit} Visit Source: [{visitSource}] {extraInformation}`

In the **E7** case which corresponds to the download of a file using Google Chrome, the URL of the downloaded file, the local path used to store it and the size of the file (amount of data already downloaded and volume of data remaining to download) are extracted using the pattern 5.

Pattern 5. `{url} ({localPath}) Received: {receivedBytes} bytes out of: {size} bytes.`

Finally, for the entry **E10** representing the deletion of a file in the recycle bin, the extraction pattern 6 is used to extract the path of the deleted file.

Pattern 6. `Deleted file: {pathFile}`

The next step consists in filtering the collected data to include only the relevant data to instantiate ORD2I. Indeed, not all data extracted from footprints are relevant for an investigation and it therefore needed to be filter them in order to reduce the amount of data to be processed by the upper layer, improve readability of the visualisation and optimise processing times. Filtering also aims to remove duplicates. The filtering was only partially implemented, but is planned for future work. Currently, our tools allow the users to filter certain data sources in order to not take them into account. However, this filtering system does not allow to manage precisely the information contained in each source (filter some attributes of a given data source while keeping some attributes of it). After filtering, the footprints are then serialised in an XML file. This aims to solve syntactic heterogeneity problems by translating data produced by the extraction layer in the format used by the upper layer. Indeed, the extraction of footprints from different sources leads to heterogeneity issues due, for example, to different formats to store dates and times (granularity, time zone, etc.). Listing 8.2 illustrates the results of the XML serialisation of events **E5**, **E7**, **E8** and **E10** (for reasons of readability, only a few events are shown).

```
<?xml version="1.0" encoding="UTF-8"?>
<extractionResults>
(E5)<footprint id="5">
  <datetime>03/04/2015 08:13:24 UTC</datetime>
  <type>Chrome History</type>
  <subtype>Webpage Visit</subtype>
  <location>WIN-LPAH04KASIA</location>
  <user>UserX</user>
  <process>Google Chrome</process>
  <description>
    <url>http://malwarewebsite.com/malwaredl/index.html</url>
    <hostname>malwarewebsite.com</hostname>
    <titleVisitedPage>Download Malware Sources and Malware
      Binaries</titleVisitedPage>
    <counterVisit>2</counterVisit>
    <isHandle>yes</isHandle>
  </description>
</footprint>
</extractionResults>
```

```

    <extra>http://malwareWebsite.com/MalwareDL/index.html (Download Malware Sources and
    Malware Binaries) [count: 2] Host: malwareWebsite.com (typed 2 times – not
    indicating directly typed count)</extra>
</footprint>
(E7)<footprint id="7">
  <datetime>03/04/2015 08:13:28 UTC</datetime>
  <type>Chrome History</type>
  <subtype>File Download</subtype>
  <location>WIN-LPAH04KASIA</location>
  <user>UserX</user>
  <process>Google Chrome</process>
  <description>
    <url>http://malwarewebsite.com/malwaredl/contagiomalware.exe</url>
    <hostname>malwarewebsite.com</hostname>
    <localPath>c:\users\userx\downloads\</localPath>
    <name>contagiomalware.exe</name>
    <receivedBytes>174054</receivedBytes>
    <sizeFile>174054</sizeFile>
    <isHandle>yes</isHandle>
  </description>
  <extra>http://malwareWebsite.com/MalwareDL/contagioMalware.exe
    (C:\Users\UserX\Downloads\contagioMalware.exe). Received: 174054 bytes out of:
    174054 bytes.</extra>
</footprint>
(E8)<footprint id="8">
  <datetime>03/04/2015 08:14:12 UTC</datetime>
  <type>Windows Prefetch</type>
  <subtype>Application Prefetch</subtype>
  <location>WIN-LPAH04KASIA</location>
  <user></user>
  <process>Windows prefetch</process>
  <description>
    <executable>contagiomalware.exe</executable>
    <runCounter>1</runCounter>
    <path>c:\users\userx\downloads\</path>
    <hash>0x82B5008B</hash>
    <volumeInfos>1 [serial number: 0xD4420B4A device path:
      \DEVICE\HARDDISKVOLUME1]</volumeInfos>
    <isHandle>yes</isHandle>
  </description>
  <extra>Prefetch [CONTAGIOMALWARE.EXE] was executed – run count 1 path:
    \USERS\USERX\DOWNLOADS\CONTAGIOMALWARE.EXE hash: 0x82B5008B volume: 1 [serial
    number: 0xD4420B4A device path: \DEVICE\HARDDISKVOLUME1] ////
    number_of_volumes: 1 volume_device_paths:
    [u'\\DEVICE\\HARDDISKVOLUME1']</extra>
</footprint>
(E10)<footprint id="10">
  <datetime>03/04/2015 08:14:47 UTC</datetime>
  <type>Recycle Bin</type>
  <subtype>File Deletion</subtype>
  <location>WIN-LPAH04KASIA</location>
  <user></user>
  <process>Recycle Bin</process>
  <description>
    <pathDeletedFile>c:\users\userx\downloads\</pathDeletedFile>
    <name>contagiomalware.exe</name>
    <driveLetter>c:</driveLetter>
    <isHandle>yes</isHandle>

```



```

    </description>
    <extra>C:\Users\UserX\Downloads\contagioMalware.exe //// file_size: 174054</extra>
  </footprint>
</extractionResults>

```

Listing 8.2: XML serialisation

Appendix D gives a comprehensive description of the information generated by the Plaso tool for each type of events of each source of information. For each of these events, the extraction rules used are given.

8.3/ INSTANTIATION

This step consists in populating the ontology using the result of the extraction process. For each *footprint* item of the XML file, the CKL and SKL are populated by creating instances of events, objects and subjects and links between individuals according to the formal properties defined in ORD2I. The relationships between an event and an object or a subject are deduced from the type of the event. For example, if a file is sent to the recycle bin, the event representing this action is linked to the file using the property *ord2i:removes*. In the case of a file download, the related event is linked to the web resource using the property *ord2i:uses* and to the local file using the property *ord2i:creates*. To illustrate the instantiation of the ontology, the mapping between the XML element corresponding to the event **E10** of Listing 8.2 and ORD2I is illustrated in Figure 8.4. Each information carries out in the XML element is transposed into the ontology in the form of concepts, object properties and datatype properties. Listing 8.3 shows a Turtle serialisation of the knowledge generated in the CKL and SKL layers during this process. The individual *ord2i:FileDeletion102* (lines 1-11) represents the event of deletion and is localised in time using the ontology *owl:Time* (lines 12-21). *ord2i:FileDeletion102* is linked to the individual *ord2i:File76*. This individual (line 25-29), belonging to *ord2i:File*, represents the file deleted by the user. *ord2i:File76* has a location represented by *ord2i:Location77* (lines 22-24) allowing to store its path and its file name. *ord2i:FileDeletion102* is linked to *ord2i:File76* using the property *ord2i:removes*. Regarding subjects, the process managing the recycle bin, used to carry out the deletion, is represented by the individual *ord2i:Process106* (line 34-38) and it is linked to the event using the property *ord2i:isInvolved*. It should be noted that the information are not sufficient to identify the user who has initiated the deletion of the file.

The knowledge describing how the previous information is identified is then added in TKL of the ontology, as shown in Listing 8.4. *ord2i:InvestigativeOperation98* (line 1-10) represents the task of information extraction performed by the tool Plaso (line 21-24). As property *ord2i:isIdentifyBy* states, this task allows to identify several entities, including *ord2i:FileDeletion102*, *ord2i:File76* and *ord2i:Process106*. To conclude, the task *ord2i:InvestigativeOperation98* is localised in time (lines 11-20).



Figure 8.4: Mapping from footprints to ORD2I

For conciseness and clarity, this chapter does not provide an exhaustive list of mapping rules.

```

1  ### http://checksem.u-bourgogne.fr/ord2i#FileDeletion102
2  :FileDeletion102 rdf:type :Event, owl:NamedIndividual ;
3    :hasEventType "Recycle Bin" ;
4    :hasEventSubtype "File Deletion" ;
5    :hasShortDescription "FileDel., contagiomalware.exe" ;
6    :hasDescription "File Deletion(Recycle Bin) made by -. File
7      contagiomalware.exe(c:\\users\\userx\\downloads\\) deleted" ;
8    :hasStatus "success" ;
9    :hasDateTime :Interval104 ;
10   :isIdentifiedBy :InvestigativeOperation98 ;
11   :hasPhysicalLocation :PhysicalLocation6 .
12   :removes :File76 ;
13   ### http://checksem.u-bourgogne.fr/ord2i#Interval104
14   :Interval104 rdf:type owl:NamedIndividual, time:Interval ;
15     time:hasDateTimeDescription :DateTimeDescription105 ;
16     time:hasBeginning :Instant103 .
17   ### http://checksem.u-bourgogne.fr/ord2i#DateTimeDescription105
18   :DateTimeDescription105 rdf:type owl:NamedIndividual, time:DateTimeDescription ;
19     time:unitType time:unitSecond .
20   ### http://checksem.u-bourgogne.fr/ord2i#Instant103
21   :Instant103 rdf:type owl:NamedIndividual, time:Instant ;
22     time:inXSDDateTime> "2015-04-03T08:14:47.000+02:00"^^xsd:dateTime .
23   ### http://checksem.u-bourgogne.fr/ord2i#PhysicalLocation6
24   :PhysicalLocation6 rdf:type :PhysicalLocation, owl:NamedIndividual ;
25     :hasMachineName "WIN-LPAH04KASIA" .
26   ### http://checksem.u-bourgogne.fr/ord2i#File76
27   :File76 rdf:type :File, owl:NamedIndividual ;
28     :hasSize "174054"^^xsd:int ;
29     :isIdentifiedBy :InvestigativeOperation98 ;
30     :hasLocation :Location77 .
31   ### http://checksem.u-bourgogne.fr/ord2i#Location77
32   :Location77 rdf:type :LocalVirtualLocation, owl:NamedIndividual ;
33     :hasPath "c:\\users\\userx\\downloads\\" ;
34     :hasFilename "contagiomalware.exe" .
35   ### http://checksem.u-bourgogne.fr/ord2i#Process106
36   :Process106 rdf:type :Process, owl:NamedIndividual ;
37     :hasName "Recycle Bin" ;
38     :isInvolved :FileDeletion102 ;
39     :isIdentifiedBy :InvestigativeOperation98 .

```

Listing 8.3: Turtle serialisation of knowledge generated by the instantiation of the CKL and SKL layers of ORD2I

```

1  ### http://checksem.u-bourgogne.fr/ord2i#InvestigativeOperation98
2  :InvestigativeOperation98 rdf:type :InvestigativeOperation , owl:NamedIndividual ;
3      :hasTruthfulness "100.0"^^xsd:double ;
4      :hasTechniqueName "Extraction using Plaso" ;
5      :hasDescription "Extraction from Recycle Bin (Recycle Bin) using Plaso" ;
6      :hasSourceName "Recycle Bin" ;
7      :hasSourceType "Recycle Bin" ;
8      :hasTechniqueType "Information Source" ;
9      :hasDateTime :Interval100 ;
10     :isPerformedWith :Tool0 .
11  ### http://checksem.u-bourgogne.fr/ord2i#Interval100
12  :Interval100 rdf:type owl:NamedIndividual , <http://www.w3.org/2006/time#Interval> ;
13      time:hasDateTimeDescription :DateTimeDescription101 ;
14      time:hasBeginning :Instant99 .
15  ### http://checksem.u-bourgogne.fr/ord2i#DateTimeDescription101
16  :DateTimeDescription101 rdf:type owl:NamedIndividual , time:DateTimeDescription ;
17      time:unitType time:unitSecond .
18  ### http://checksem.u-bourgogne.fr/ord2i#Instant99
19  :Instant99 rdf:type owl:NamedIndividual , time:Instant ;
20      time:inXSDDateTime "2015-06-19T09:46:49.000+02:00"^^xsd:dateTime .
21  ### http://checksem.u-bourgogne.fr/ord2i#Tool0
22  :Tool0 rdf:type :Tool , owl:NamedIndividual ;
23      :hasName "Plaso" ;
24      :hasVersion "1.2.0" .

```

Listing 8.4: Turtle serialisation of knowledge generated by the instantiation of the TKL layer of ORD2I

8.4/ ANALYSIS

After the instantiation of the ontology, a large knowledge graph representing the information contained in the output of Plaso is obtained. The structure of this graph is derived from the schema of our ontology. This graph presents many advantages because it structure information and it facilitates the interpretation of information and the conception of automatic analysis processes. As mentioned in Chapter 3, the analysis of this large volume of knowledge requires the use of sophisticated tools. In this section, we present the inference tools and automated analysis processes proposed by our approach. It is important to note that the proposed operators are based on assumptions made by the authors that can be subject to discussion. However, the main objective is to demonstrate the relevance of an ontology as this greatly facilitates the development of the analysis tools.

8.4.1/ KNOWLEDGE ENHANCEMENT

The goal of the enhancement step is to enrich the knowledge base with new deduced facts in order to complete the investigators' knowledge of the incident. The inference of

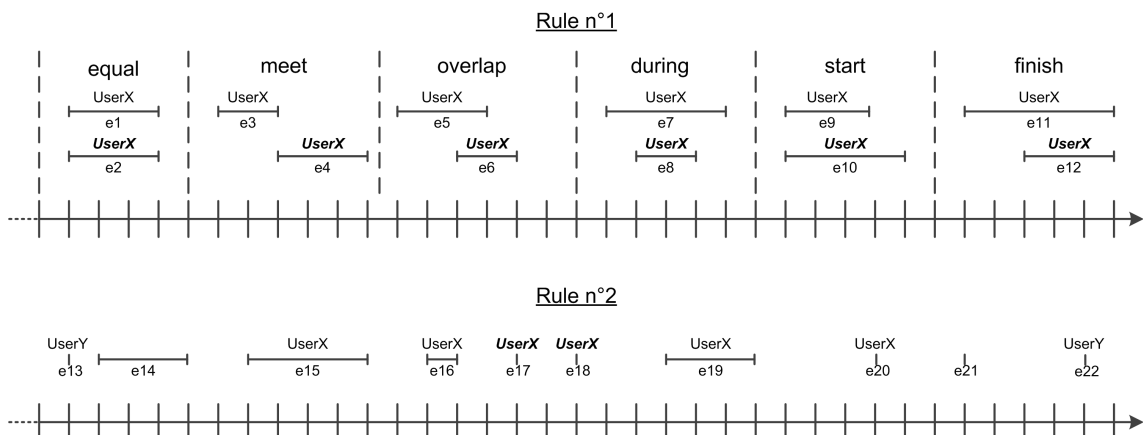


Figure 8.5: Patterns used to deduce new relationships between users and events

new facts increases the effectiveness of the analysis and the quality of final results. Indeed, adding new knowledge about the incident during the enhancement stage improves the quality of the conclusions. The inference is an operation taking as input one or more established facts and outputting a new fact, called conclusion, that was implicit before the deduction. In our work, we have implemented several inference rules to illustrate the capabilities of this mechanism. Two examples are given below.

8.4.1.1/ DEDUCTION OF THE INVOLVEMENT OF A USER IN AN EVENT

The first example of inference is the deduction of the OS user session associated with a given event when this information is unknown. Any information on the user session associated with an event is valuable for an investigation. Certain types of events provide such information while others not. This is true for the entries from *Windows EVT*X for which the user field is filled in the output of Plaso. This is also the case of entries generated by web browsers for which the user name is contained in the path of history files used to generate the entry (i.e. in the case of Google Chrome, every history file is located in `TSK:/Users/UserX/AppData/Local/Google/Chrome/UserData/Default/History`). To identify the user associated with an event, we introduce a tool to identify this information when it is not directly available in the footprints. We define two inference rules which are based on the temporal location of events defined using Allen algebra (Allen, 1983) and the notion of time interval. Our ontology represents the beginning and the end of each event to model the duration of the latter (i.e. downloading a file that lasts for several minutes). However, because of the granularity of the timestamps and the execution speed, most of the events are considered instantaneous. In addition, the end of the event is not often available in the output of Plaso. In most cases, therefore, we consider that each event is localised in time by an interval whose have a start time and an end time equal to the datetime provided by Plaso.

The two inference rules used to identify the involvement of users are:

- Rule №1: Given an event A for which the user is unknown and an event B associated to the user P , if A and B are temporally linked by a property $p \in \{equal, overlaps, during, starts, meets, finishes\}$, then the user associated to A is P .
- Rule №2: Given two events A and B associated to the user P and located at both extremities of a chain of events exclusively made of events for which the user is unknown, then each event composing this chain is associated to the user P .

Event	Time	User
ord2i:FileContentModification5 (E1)	2015-04-03T08:13:02	
ord2i:ApplicationPrefetch17 (E2)	2015-04-03T08:13:13	
ord2i:CookieUse28 (E3)	2015-04-03T08:13:14	ord2i:Person33
ord2i:FileCreation40 (E4)	2015-04-03T08:13:16	ord2i:Person33
ord2i:WebpageVisit50 (E5)	2015-04-03T08:13:24	ord2i:Person33
ord2i:WebpageVisit60 (E6)	2015-04-03T08:13:27	ord2i:Person33
ord2i:FileDownload70 (E7)	2015-04-03T08:13:28	ord2i:Person33
ord2i:ApplicationPrefetch82 (E8)	2015-04-03T08:14:12	ord2i:Person33
ord2i:WebpageVisit92 (E9)	2015-04-03T08:14:37	ord2i:Person33
ord2i:FileDeletion102 (E10)	2015-04-03T08:14:47	
ord2i:FileCreation111 (E11)	2015-04-03T08:15:29	
ord2i:FileAccess121 (E12)	2015-04-03T08:15:51	ord2i:Person33

Table 8.1: Identification of the involvement of users using knowledge enhancement

These two rules are illustrated in Figure 8.5 (the user in bold italic is the result of the enhancement). The upper part of the figure illustrates the rule №1. This rule is used to deduce the user associated with the events e2, e4, e6, e8, e10, e12. The bottom part of the figure illustrates the rule №2 and shows that the UserX is involved in e17 and e18.

After the instantiation of the ontology using as input the entries given in Listing 8.1, the knowledge we have about users involved in each event can be retrieved using the SPARQL query given in Listing 8.5.

```

PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX time: <http://www.w3.org/2006/time#>
PREFIX ord2i: <http://checksem.u-bourgogne.fr/ord2i#>
SELECT ?event ?startTime ?user
WHERE
{
  ?event a ord2i:Event .
  ?event ord2i:hasDateTime ?time .
  ?time time:hasBeginning ?begin .
  ?begin time:inXSDDateTime ?startTime .
  OPTIONAL
  {
    ?user ord2i:isInvolved ?event .
    ?user a ord2i:Person .
  }
}

```

ORDER BY ?startTime

Listing 8.5: SPARQL query retrieving information about the user involved in each event

The column *user* in Table 8.1 shows the relation between user and event. The users with normal font are known directly after the instantiation of the ontology (as the information can be retrieved in the output of Plaso) while the users appearing in bold italic are deduced during the enhancement phase. After completing the enhancement phase and executing the SPARQL query again, a link between *ord2i:ApplicationPrefetch82* and *ord2i:Person33* is identified by rule №2. As the user *ord2i:Person33* is involved in *ord2i:FileDownload70* and *ord2i:WebpageVisit92* and that *ord2i:ApplicationPrefetch82* is located between these two events, the system states that *ord2i:Person33* is also involved in this event.

```

1  ### http://checksem.u-bourgogne.fr/ord2i#ApplicationPrefetch82
2  :ApplicationPrefetch82 rdf:type :Event, owl:NamedIndividual ;
3    :hasEventType "Windows Prefetch" ;
4    :hasEventSubtype "Application Prefetch" ;
5    :hasDescription "Application Prefetch (Windows Prefetch) made by -. Application
6      contagiomalware.exe(c:\\users\\userx\\downloads\\) added in Prefetch folder" ;
7    :hasShortDescription "Prefetch, contagiomalware.exe" ;
8    :hasStatus "success" ;
9    :hasDateTime :Interval84 ;
10   :isIdentifiedBy :InvestigativeOperation129, :InvestigativeOperation78 ;
11   :hasPhysicalLocation :PhysicalLocation6 ;
12   :uses :WinExeFile86 .
13  ### http://checksem.u-bourgogne.fr/ord2i#Person33
14  :Person33 rdf:type :Person, owl:NamedIndividual ;
15    :hasPseudo "UserX" ;
16    :isInvolved :CookieUse28, :FileAccess121, :FileCreation40, :FileCreation111 ;
17    :isInvolved :WebpageVisit50, :WebpageVisit60, :WebpageVisit92 ;
18    :isInvolved :ApplicationPrefetch82, :FileDownload70, :FileDeletion102 ;
19    :isRelatedTo :WinUserAccount34 .
20  ...
21  ### http://checksem.u-bourgogne.fr/ord2i#InvestigativeOperation129
22  :InvestigativeOperation129 rdf:type :InvestigativeOperation, owl:NamedIndividual ;
23    :hasTruthfulness "75.0"^^xsd:double ;
24    :hasTechniqueName "User Finder" ;
25    :hasTechniqueType "Knowledge Enhancement" ;
26    :hasDescription "User Finder using rule n 2 : Let two events A and B, linked with a
27      user P, located at both extremities of a chain of events for which the user is
28      unknown, then all events composing this chain are linked with P." ;
29    :isSupportedBy :FileDownload70 ;
30    :isSupportedBy :WebpageVisit92 .
31    :hasDateTime :Interval131 ;
32    :isPerformedWith :Tool127 ;

```

Listing 8.6: Knowledge inserted in ORD2I for a deduction

Listing 8.6 shows the knowledge about *ord2i:ApplicationPrefetch82* (lines 1-11) after the enhancement. A new investigative operation, representing the operation of deduction, is added to the ontology (*ord2i:InvestigativeOperation129* (lines 20-29)). This investigative operation infers that *ord2i:Person33* (lines 12-19) is now involved (*ord2i:isInvolved*) in *ord2i:ApplicationPrefetch82*. It should be noted that the inference of new associations between users and events cannot be made with certainty. Indeed, another user can take the place of the current user or the events can be initiated by a person or a process beyond the control of the user who opened the session. Thus, the score of confidence associated with the operation that has deduced the association is low to invite the investigators to use this information cautiously. This means that the value associated to the datatype property *ord2i:hasTruthfulness* of the instance belonging to *ord2i:InvestigativeOperation* and corresponding to this operation is set to a low value.

8.4.1.2/ FILE CLASSIFICATION

The second inference operator used in our approach allows to classify automatically the files. During the extraction, it is sometimes impossible to determine the type of a file. When such a resource is extracted from the digital crime scene, an instance of the *ord2i:File* class is automatically created during the instantiation of the ontology. To characterise more precisely these resources, an inference operator is used to classify the corresponding instances into the classes belonging to *ord2i:File* (e.g. *ord2i:ExeFile*, *ord2i:ArchiveFile*, *ord2i:ImageFile*, *ord2i:PDFFile*, etc.). To classify the instances, the type of the file is deduced based on the extension of the file contained in its filename. For example, after the extraction step, three resources named report.pdf, setup.exe and data.rar are identified. Three instances belonging to *ord2i:File* are then created during the instantiation phase to model this knowledge. During the enhancement phase, these three instances are classified in the subclasses of *ord2i:File* to represent more accurately the real nature of these resources. In this case, the instances are declared respectively as instance of the classes *ord2i:PDFFile*, *ord2i:ExeFile* and *ord2i:ArchiveFile*.

8.4.2/ TIMELINE ANALYSIS

The first analysis tool proposed in our approach is a process (based on Section 6.2.2) allowing to detect correlation between a pair of events. The relevance of this tool will be shown in Chapter 9. As described in Chapter 6, the identification of correlated pairs of events is performed using four criteria:

- $Corr_T(e_1, e_2)$ is a score quantifying the temporal correlation of the two events. The hypothesis used in this criterion is that if two events occur at the same time, the temporal correlation is equal to 1.0. Else, the more the two events are closed in time, the more they are correlated.
- $Corr_S(e_1, e_2)$ is a score quantifying the correlation in the light of subjects interacting with the two events. The more the two events interact with common subjects (process or person), the greater the subject correlation is. Therefore, the subject correlation is maximal for two events interacting with only two subjects, the Google Chrome process and the same user for example. In the case of two events interacting with the Google Chrome process but with two different users, the score is equal to 0.5.
- $Corr_O(e_1, e_2)$ is a score quantifying the correlation in the light of objects interacting with the two events. The object correlation increases when the two events interact with similar objects. To obtain a finer system, we choose to not only consider common objects but also objects whose location is near as explain in Section 6.2.2 of Chapter 6. For example, in the case of two events interacting with two objects sharing the same hostname (in case of remote resources) or the same folder (in

case of local resources), the score is increased.

- $Corr_{KBR}(e_1, e_2)$ is a score quantifying the correlation using rules based on expert knowledge (i.e. knowledge defined by forensic expert).

For each of these criteria, a score between 0.0 and 1.0 (after normalisation) is computed. Each criterion can be weighted to allow to give more importance to one of the correlation score (in this study, all scores are equivalently weighted). The overall correlation score is equal to a score between 0.0 (meaning that the two events are not correlated) and 1.0 (meaning that the two events are strongly correlated), which is the average of the three scores. The three first criteria are not dependent on the type of events as they are based on generic features (time, object and subjects). This will allow to discover correlations involving events from unknown sources of information that are therefore not intended by the investigators. To take into account specific knowledge of every type of object model in ORD2I, we introduce a fourth score $Corr_{EK}(e_1, e_2)$. This score is computed using a set of rules defined by experts (called expert knowledge-based rules). If two events satisfy a certain given rule, the overall correlation score between them is equal to the score associated with this rule. For example, in our experiments, we add the following two rules: The first rule is used to get a high correlation score (1.0) in the case of two events; one representing the download of an executable file and the second representing its execution. The second rule allows to get a high correlation score (1.0) in a case such as an event representing the creation of a bookmark for a webpage and an event representing a visit of this same webpage using the bookmark.

To avoid calculating the correlation score of all pairs of events and thus improve the performance of the system, a windowing mechanism is used. The investigator has the possibility, through the graphical interface of the tool, to define a time correlation window which is then used to limit the correlation of events only for pairs of events being in the scope of the window. This sliding window has a length depending on the choice of the investigator (i.e. 15min, 30min, 1h, 6h, 12h, 24h or no window). The investigator also has the possibility to define a threshold for each score computed in addition to the overall score. These thresholds aim to keep significant correlations only.

To illustrate the correlation process, we use our approach on the dataset made of seven events shown in Listing 8.1. The story of this dataset appears to be the following: First, a file was modified by the system (*ord2i:FileContentModification5*) (**E1**). The first action of the user was to start the web browser Google Chrome (*ord2i:ApplicationPrefetch17*) (**E2**). A cookie was then used (*ord2i:CookieUse28*) (**E3**) and a cache file is created (*ord2i:FileCreation40*) (**E4**). Then, the user visited two webpages (*ord2i:WebpageVisit50*) (**E5**) and *ord2i:WebpageVisit60* (**E6**) on a website that seem to provide sources and binaries of malware. On the second webpage, he found a link allowing him to download an executable file named *contagiomalware.exe* (*FileDownload70*) (**E7**). Once the download was complete, the user ran the executable (*ord2i:ApplicationPrefetch82*) (**E8**). He continued to browse the web on a non

related website (*ord2i:WebpageVisit92* (**E9**)). The user then deleted the file *contagiom malware.exe* (*ord2i:FileDeletion102* (**E10**)). Finally, two files were respectively created (*ord2i:FileCreation111* (**E11**)) and accessed (*ord2i:FileAccess121* (**E12**)). Among these events, it is possible to identify a logical chain of events using intuition. This chain is composed of the following events *ord2i:WebpageVisit50*, *ord2i:WebpageVisit60*, *ord2i:FileDownload70*, *ord2i:App.Prefetch82* and *ord2i:FileDeletion102*. These events make up a coherent story in which the user gets a malware provided by a website, executes it and attempts to erase the traces by deleting the file.

The main objective of the proposed correlation tool is to highlight this type of chain of events among a large number of events composing a case. Thus, on one hand, one can understand the case quickly and on the other hand see the whole chain of events in which a given event has played a role. The most significant correlations (overall score > 0.4) are given in Table 8.2.

Nº	evt1	evt2	Temp.	Subj.	Obj.	EK	Overall
1	<i>App.Prefetch82</i> (E8)	<i>FileDownload70</i> (E7)	0.01	0.5	1.0	1.0	1.0
2	<i>FileDownload70</i> (E7)	<i>WebpageVisit60</i> (E6)	1.0	1.0	0.25	0.0	1.0
3	<i>App.Prefetch82</i> (E8)	<i>FileDeletion102</i> (E10)	0.02	0.5	1.0	0.0	0.85
4	<i>FileDeletion102</i> (E10)	<i>ord2i:FileDownload70</i> (E7)	0.01	0.5	1.0	0.0	0.84
5	<i>WebpageVisit50</i> (E5)	<i>WebpageVisit60</i> (E6)	0.33	1.0	0.25	0.0	0.72
6	<i>FileDownload70</i> (E7)	<i>WebpageVisit50</i> (E5)	0.25	1.0	0.25	0.0	0.68
7	<i>CookieUse28</i> (E3)	<i>WebpageVisit50</i> (E5)	0.09	1.0	0.0	0.0	0.46
8	<i>CookieUse28</i> (E3)	<i>WebpageVisit60</i> (E6)	0.07	1.0	0.0	0.0	0.45
9	<i>CookieUse28</i> (E3)	<i>FileDownload70</i> (E7)	0.07	1.0	0.0	0.0	0.45
10	<i>FileAccess121</i> (E12)	<i>FileCreation111</i> (E11)	0.04	1.0	0.0	0.0	0.44
11	<i>FileDownload70</i> (E7)	<i>WebpageVisit92</i> (E9)	0.01	1.0	0.0	0.0	0.42
12	<i>WebpageVisit60</i> (E6)	<i>WebpageVisit92</i> (E9)	0.01	1.0	0.0	0.0	0.42
13	<i>WebpageVisit50</i> (E5)	<i>WebpageVisit92</i> (E9)	0.01	1.0	0.0	0.0	0.42
14	<i>CookieUse28</i> (E3)	<i>WebpageVisit92</i> (E9)	0.01	1.0	0.0	0.0	0.42
15	<i>FileCreation111</i> (E11)	<i>FileCreation40</i> (E4)	0.0	1.0	0.0	0.0	0.42
16	<i>FileAccess121</i> (E12)	<i>FileCreation40</i> (E4)	0.0	1.0	0.0	0.0	0.42
17	<i>App.Prefetch17</i> (E1)	<i>CookieUse28</i> (E3)	1.0	0.0	0.0	0.0	0.42
18	<i>CookieUse28</i> (E3)	<i>FileCreation40</i> (E4)	0.5	0.5	0.0	0.0	0.41

Table 8.2: Correlation scores for events from Listing 8.1 (threshold > 0.4)

This table contains nine pairs of correlated events for which five scores are given: the temporal correlation, the subject correlation, the object correlation, the correlation based on rules and finally the overall correlation. The scores show six pairs highly correlated (*overallscore* > 0.5):

- Correlation Nº1 (score=1.0): *ord2i:ApplicationPrefetch82* and *ord2i:FileDownload70* satisfied the rule defined below ("download of an executable file and execution of the same file").
- Correlation Nº2 (score=1.0): The subject correlation between *ord2i:FileDownload70* and *ord2i:WebpageVisit60* is maximal as they are both related to only two subjects, Google Chrome and the user Person22. In addition, as only one second has elapsed between the visit of the webpage containing the download link of *contagiom malware.exe* and the start of the download, the temporal correlation is high too.

Finally, as the URL of the remote resource used for the download and the URL of the visited webpage are hosted by the same website, the object correlation is modified accordingly.

- Correlation №3 (score=0.85): The correlation between *ord2i:ApplicationPrefetch82* and *ord2i:FileDeletion102* can be explained by the interaction with the same object, one event downloading an object and the other event deleting the same object.
- Correlation №4 (score=0.84): The same explanation can be applied to the correlation between *ord2i:FileDeletion102* and *ord2i:FileDownload70*. For these two last correlation scores, the subject correlation score and the temporal correlation score are low as they do not interact with common subjects and they are not close in time (compared to other events).
- Correlation №5 (score=0.72): *ord2i:WebpageVisit50* and *ord2i:WebpageVisit60*: the object and subject correlation scores can be explained in the same way as for the correlation between *ord2i:FileDownload70* and *ord2i:WebpageVisit60*.
- Correlation №6 (score=0.68): *ord2i:FileDownload70* and *ord2i:WebpageVisit50*: the same explanation can be used for this case too.

All significant correlations (correlation with an overall score greater than the threshold) are then added to the knowledge base in order to avoid calculating them again if investigators need to consult one of them again. To illustrate the representation of a correlation in the base, the knowledge added for the correlation of events *ord2i:ApplicationPrefetch82* and *ord2i:FileDownload70* is given in Listing 8.7.

```

1  ### http://checksem.u-bourgogne.fr/ord2i#Correlation206
2  :Correlation206 rdf:type :Correlation, owl:NamedIndividual ;
3      :hasTemporalCorrelation "0.016910171"^^xsd:float ;
4      :hasSubjectCorrelation "0.5"^^xsd:float ;
5      :hasOverallCorrelation "1.0"^^xsd:float ;
6      :hasEKBasedCorrelation "1.0"^^xsd:float ;
7      :hasObjectCorrelation "1.0"^^xsd:float ;
8      :hasCorrelatedEvent :ApplicationPrefetch82, :FileDownload70 ;
9      :isIdentifiedBy :InvestigativeOperation207 .
10 ### http://checksem.u-bourgogne.fr/ord2i#InvestigativeOperation207
11 :InvestigativeOperation207 rdf:type :InvestigativeOperation, owl:NamedIndividual ;
12     :hasTruthfulness "50.0"^^xsd:double ;
13     :hasTechniqueType "Analysis" ;
14     :hasDescription "Event Correlation" ;
15     :hasTechniqueName "Event Correlation" ;
16     :isSupportedBy :ApplicationPrefetch82, :FileDownload70 ;
17     :hasDateTime :Interval209 ;
18     :isPerformedWith :Tool145 .

```

Listing 8.7: Knowledge about the correlation between *ord2i:ApplicationPrefetch82* and *ord2i:FileDownload70*

For each significant correlation (scores greater or equal than the threshold), a new individual belonging to the *ord2i:Correlation* is added to the knowledge base (*ord2i:Correlation206* (lines 1-9)). This cor-

relation carries the correlation scores using the datatype properties *ord2i:hasSubjectCorrelation*, *ord2i:hasObjectCorrelation*, *ord2i:hasTemporalCorrelation*, *ord2i:hasEKBasedCorrelation* and *ord2i:hasOverallCorrelation*. The correlated events *ord2i:ApplicationPrefetch82* and *ord2i:FileDownload70* are pointed by the correlation using the object property *ord2i:hasCorrelatedEvent*. An individual is also added in TKL to model the task that found this correlation (*ord2i:InvestigativeOperation207* (lines 10-18)).

8.5/ DATA VISUALISATION

The interface layer has a number of functions to:

- Visualise the data via an intuitive and clear visualisation tool.
- Provide a monitoring interface allowing to manage the settings of the system, including the numerical values used in the correlation process (weight, size of the sliding window, etc.) and the information sources at the input of the instantiation step.
- Provide an advanced query tool for users that existing visualisation tools are not able to perform.

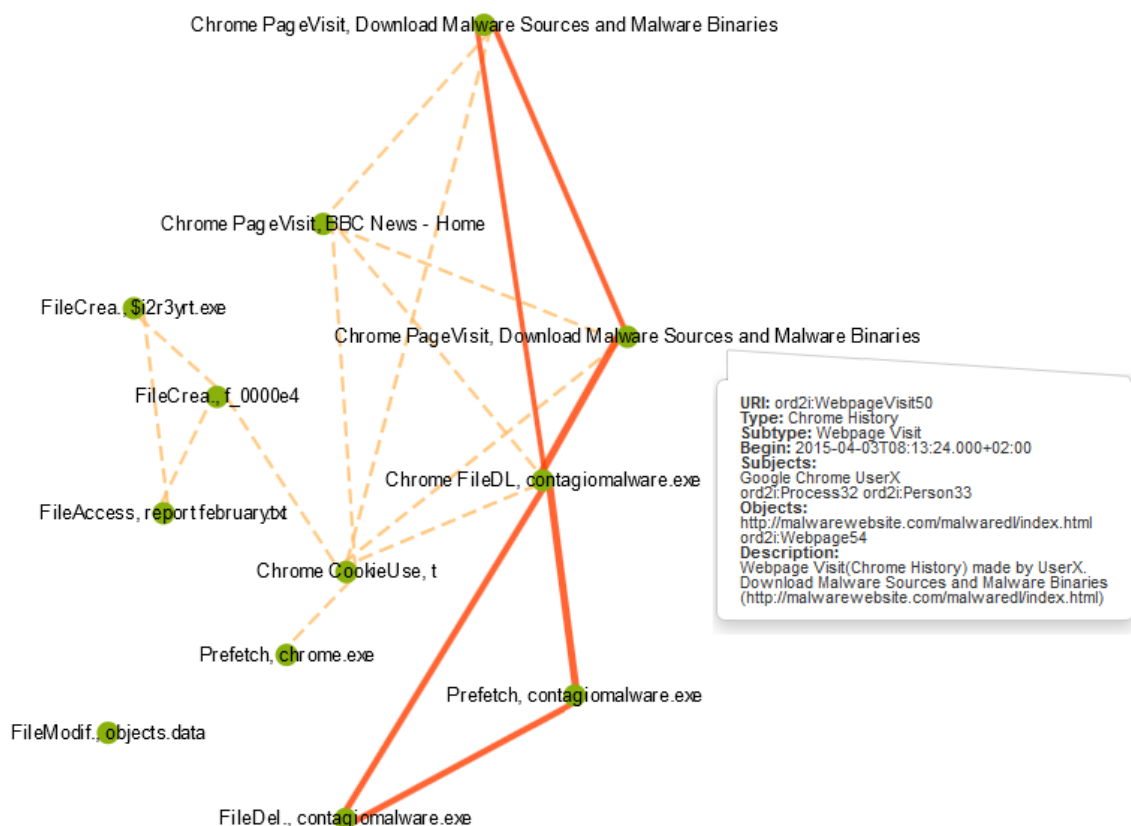


Figure 8.6: Correlation graph of events given in Listing 8.1 (solid edges (*score* \geq 0.6), dashed edges (*score* \geq 0.4 && *score* $<$ 0.6))

A tool has been implemented to visualise event correlation. This visualisation takes the form of a graph containing all the events composing the incident and the correlations between them. The use of this graph allows to quickly have an overview of the case. The links between events show chains of related events. The visual style (colour, line type, thickness) of each link depends on the strength of the correlation. The investigator can interact with the graph by clicking on elements. A click on a link gives information about the correlation such as the overall correlation score, the temporal correlation score, the subject correlation score, the object correlation score and the correlation score based on rules. A click on a node gives information about the event (URI, description, objects and subjects interacting with the event). The collection of this information from the triple store is done with the dynamic creation of SPARQL queries.

An overview of the correlation graph corresponding to the dataset of Listing 8.1 is given in Figure 8.6. There appears a chain of strongly correlated events including the visit of the webpage, the download of the resource, the launch of this resource and its removal. With this graph, we can identify the following chain of events *ord2i:WebpageVisit50*, *ord2i:WebpageVisit60*, *ord2i:FileDownload70*, *ord2i:App.Prefetch82* and *ord2i:FileDeletion102* discussed above.

To go further and display information which does not appear in the visualisation tool, one can use the SPARQL endpoint to query directly the knowledge contained in the triple store. The SPARQL language coupled with the expressiveness of the language OWL 2 (and by extension, RDF/RDFS) allows to query the knowledge base in an intuitive way and to quickly obtain the desired knowledge. For example, a query retrieving information about all webpages visited by the user named UserX is given in Listing 8.8.

```

PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX time: <http://www.w3.org/2006/time#>
PREFIX ord2i: <http://checksem.u-bourgogne.fr/ord2i#>
SELECT ?titleWebpage ?urlWebpage ?hostWebpage
WHERE
{
  ?event rdf:type ord2i:Event.
  ?event ord2i:hasEventSubtype "Webpage Visit".
  ?event ord2i:uses ?object.
  ?object ord2i:hasTitle ?titleWebpage.
  ?object ord2i:hasLocation ?location.
  ?location ord2i:hasURL ?urlWebpage.
  ?location ord2i:hasHostname ?hostWebpage.
  ?user ord2i:isInvolved ?event.
  ?user ord2i:hasPseudo ?pseudo.
  FILTER(?pseudo = "UserX").
}

```

Listing 8.8: SPARQL query retrieving information about all webpages visited by the user named UserX

The results of this query are given in Table 8.3.

titleWebpage	hostWebpage
Download Malware Sources and Binaries	malwarewebsite.com
Download Contagio	malwarewebsite.com
BBC News - Home	www.bbc.co.uk

Table 8.3: Results of the query retrieving information about all webpages visited by the user named UserX

The user can also get information about the provenance of each information contained in the knowledge base. For this, the SPARQL endpoint is used to query the knowledge contained in the TKL.

For example, the user can get an overview of what have been extracted from the disk image by Plaso using the query given in Listing 8.9 (the prefixes are hidden for readability purpose). This query returns each event that has been identified by Plaso. For each of this event, the date of the extraction and the information source used are given.

```

SELECT ?event ?date ?source
WHERE
{
  ?event a ord2i:Event.
  ?event ord2i:isIdentifiedBy ?invOp.
  ?invOp ord2i:isPerformedWith ?tool.
  ?invOp ord2i:hasDateTime ?dateTime.
  ?invOp ord2i:hasSourceType ?source.
  ?dateTime time:hasBeginning ?begin.
  ?begin time:inXSDDateTime ?date.
  ?tool ord2i:hasName ?toolName.
  FILTER(?toolName = "Plaso")
}

```

Listing 8.9: SPARQL query giving the list of the events extracted using Plaso

The results of this query are given in Table 8.4.

event	date	source
ord2i:FileContentModification5	2015-08-18T12:07:53.000+02:00	File System
ord2i:ApplicationPrefetch17	2015-08-18T12:07:53.000+02:00	Windows Prefetch
ord2i:FileCreation40	2015-08-18T12:07:54.000+02:00	File System
ord2i:FileDownload70	2015-08-18T12:07:54.000+02:00	Chrome History
ord2i:WebpageVisit92	2015-08-18T12:07:54.000+02:00	Chrome History
ord2i:FileCreation111	2015-08-18T12:07:54.000+02:00	File System
ord2i:FileAccess121	2015-08-18T12:07:54.000+02:00	File System
ord2i:WebpageVisit50	2015-08-18T12:07:54.000+02:00	Chrome History
ord2i:WebpageVisit60	2015-08-18T12:07:54.000+02:00	Chrome History
ord2i:ApplicationPrefetch82	2015-08-18T12:07:54.000+02:00	Windows Prefetch
ord2i:CookieUse28	2015-08-18T12:07:53.000+02:00	Firefox History
ord2i:FileDeletion102	2015-08-18T12:07:54.000+02:00	Recycle Bin

Table 8.4: Results of the query giving the list of all the events extracted using Plaso

Regarding traceability, the SPARQL endpoint can also be used to get explanations about the correlation between two events (i.e. get the scores used to compute the overall correlation) or to know how a given fact was deduced and what knowledge were used to make this deduction.

Other examples of the use of SPARQL are given in Chapter 9. Despite its power, the SPARQL language (as SQL) is not an intuitive and simple way to access knowledge and it requires technical knowledge to be used efficiently. Therefore, we have the desire to minimise the use of SPARQL by integrating a large amount of data in the visualisation tools. Another possible improvement is to facilitate the creation of SPARQL queries using tools like Sparklis (Ferré, 2014). This faceted search tool provides an interface inviting users to express their queries using natural language. Each query is then automatically translated into SPARQL. Therefore, this tool allows to council the expressiveness and the readability of natural language with the scalability of SPARQL.

8.6/ CONCLUSION

This chapter has presented a software architecture, named ANNALIST, providing tools to carry out the reconstruction of digital incidents. This architecture is based on the theoretical foundations developed in Chapter 6 and is centred on the ORD2I ontology defined in Chapter 7. ANNALIST demonstrates the technical feasibility of the SADFC approach. Among the tools provided by this architecture, the analysis and visualisation based on the ontology appear to be promising to investigate efficiently on a forensic case. These features help to save time by identifying and displaying on the screen the correlation relationships between events. This will help to quickly understand the physiognomy of the incident. To illustrate the capabilities of the architecture and the relevance of its features, Chapter 9 provides a quantitative study and a qualitative study of it.

EXPERIMENTS

This chapter aims to evaluate the tools proposed by the SADFC approach. The evaluation must take into account several aspects. A digital forensics tool needs to produce precise and accurate results while meeting the legal requirements and the temporal constraints fixed by the court and the laws. First, the precision quantifies the ability of the tools to generate correct results. This means the capacity of the tools to draw conclusions that prove to be accurate. Second, as said in Section 2.4 of Chapter 2, every forensic tool has to meet several legal requirements. These requirements concern the credibility of the results, their reliability and their reproducibility. This chapter pays particular attention to show that the proposed tools have the capacity to explain the results produced. Finally, the results have to be generated in a time frame consistent with the deadlines set in the framework of the investigation. Therefore, the tools must have good performance. This chapter evaluates the ability of the tools composing ANNALIST to deliver the results in a timely manner. It also assesses their capacity to process large volumes of data to meet the needs of real investigations.

This chapter is structured in the following way. Section 9.1 presents the data sets used for the evaluation. In this section, several fictitious data sets are proposed and are characterised (types of information sources in the data set, quantity of entries for each type, etc.). Section 9.2 is a quantitative evaluation which assesses the performance of the approach. This section evaluates the execution time of each step composing the process implemented by ANNALIST and the amounts of data processed, generated and inferred during every step. This study also assesses the ability of the solution to process heterogeneous sources. Section 9.3 is a qualitative evaluation of the approach assessing the accuracy and the reproducibility of the results through case study. This section also aims to demonstrate the relevance of SADFC showing how it can be used to facilitate the work of the investigators. In particular, examples are provided to show how the investigations can be conducted using the tools provided by the ANNALIST architecture (SPARQL endpoint, reasoning tools, data visualisation, etc.).

9.1/ DATA SETS

For comparison purposes, three data sets of different sizes are studied in this chapter. Disk images used are generated from a virtual machine running Windows 7 and are in the EnCase format. Their size is between 3GB and 10GB. It should be noted that the files used as input of the ANNALIST architecture are fragments of these disk images. Indeed, a disk image contains many events that can be filtered from the beginning because they are related to starting or stopping the machine and they are therefore usually not relevant for the investigation. The composition of the three datasets is given in Table 9.1. This table gives the number of events for each type of events that can be collected from the image. The data sets studied contains 9 types of events that can be categorised into 23 subtypes. To get precisions about each types of events, the reader can refer to the Appendix D.

Type of events	Data set №1	Data set №2	Data set №3
FILE	3615	6006	3942
—Modification	1294	1797	1786
—Access	511	30	24
—Creation	616	1349	60
—Birth	1194	2830	2072
REGISTER	321	2265	3432
—SYSTEM Key	35	902	89
—SOFTWARE Key	69	454	243
—NTUSER key	149	541	185
—SAM key	2	1	2
—SECURITY key	0	1	0
—UNKNOWN key	71	366	2913
EVT	243	1406	360
—Win EVTX	243	1406	360
LOG	30	13	33
—Win Preftech	30	13	33
WEB HISTORIES	4430	6779	13623
—Chrome Cache	3120	6	14
—Chrome Cookies	1006	1006	1777
—Chrome History	176	130	727
—Firefox Cache	0	2231	0
—Firefox Cookies	0	0	0
—Firefox History	0	246	0
—Internet Explorer Cache File	128	3160	11105
LINK	9	31	17
—Windows Shortcut	9	31	17
RECYCLE BIN	6	3	0
OLECF	3	8	15
JOB	2	2	2
TOTAL	8664	16513	21424

Table 9.1: Description of the data sets: types of events and numbers of events for each type

9.2/ QUANTITATIVE EVALUATION

The first task of the experiment is to evaluate the execution time of our tool and the volume of data handled and generated through the process. The configuration of the machine used to run the experiment and hosting the triple store (Stardog server 2.2.4) has a 3.20GHz Intel Core i5-3470 processor and 8GB RAM.

Table 9.2 provides information including the number of entries composing the timeline, the number of entries supported by our tool (which do not support all sources of information), the number of entries after filtering, the number of triples generated during the instantiation phase, the number of triples deducted during the enhancement phase and finally the number of significant correlations found ($score > 0.5$, this threshold can be modified by the user via the interface). The number of entries is reduced by using some filtering mechanism during the instantiation phase. These filters are to used only the entries of certain sources of information. Thus, entries from Chrome cookies, Chrome cache, Internet Explorer cache and the file system are filtered as we argue that these sources are rarely essential information to investigate. It is important to see that the number of triples does not increase linearly with the number of entries in the timeline. Indeed, the number of triples generated for an event depends on its type as some types of events carry more information.

Criterion \ Data sets	Data set №1	Data set №2	Data set №3
Number of lines (timeline)	8664	16513	21424
Supported entries (extraction)	7918	14104	17863
Entries after filtering	222	416	774
Generated triples (instantiation)	10186	17845	33498
Deducted triples (enhancement)	21	17	28
Correlations (≥ 0.5)	1411	2268	15356

Table 9.2: Quantification of processed data volumes through the reconstruction and analysis process

Steps \ Data sets	Data set №1	Data set №2	Data set №3
Extraction	1.2	1.7	1.9
Instantiation	26.8	50.9	186.3
Enhancement	0.3	0.3	0.35
Analysis	814.2	4313.5	13541.8

Table 9.3: Execution times of the different stages of the process

Table 9.3 gives the execution times (in seconds) for the different phases of the process. The curves in Figure 9.1 show a comparison of the execution times observed for the three different data sets. From these results, we can see that the extraction, instantiation and enhancement phases are carried out quickly. The blue curve shows that the extraction step is extremely fast and almost constant between data sets. The same observation applies to the enhancement step (blue curve). For this step, the low execution time is mainly due to the low number of rules currently used to make deductions. As shown in

Table 9.2, currently, few facts are produced by this step. On its side, the execution time of the instantiation step (red curve) increases with the amount of data. This increase is legitimate given that the amount of triplets to be inserted in the knowledge base increases with the size of the input, as shown in Table 9.2. Finally, the analysis step (purple curve) is the most time consuming task. Furthermore, one can see a large increase of the execution time for larger data volumes. This emphasises the importance of filtering the sources of non-critical information to reduce the amount of data to analyse.

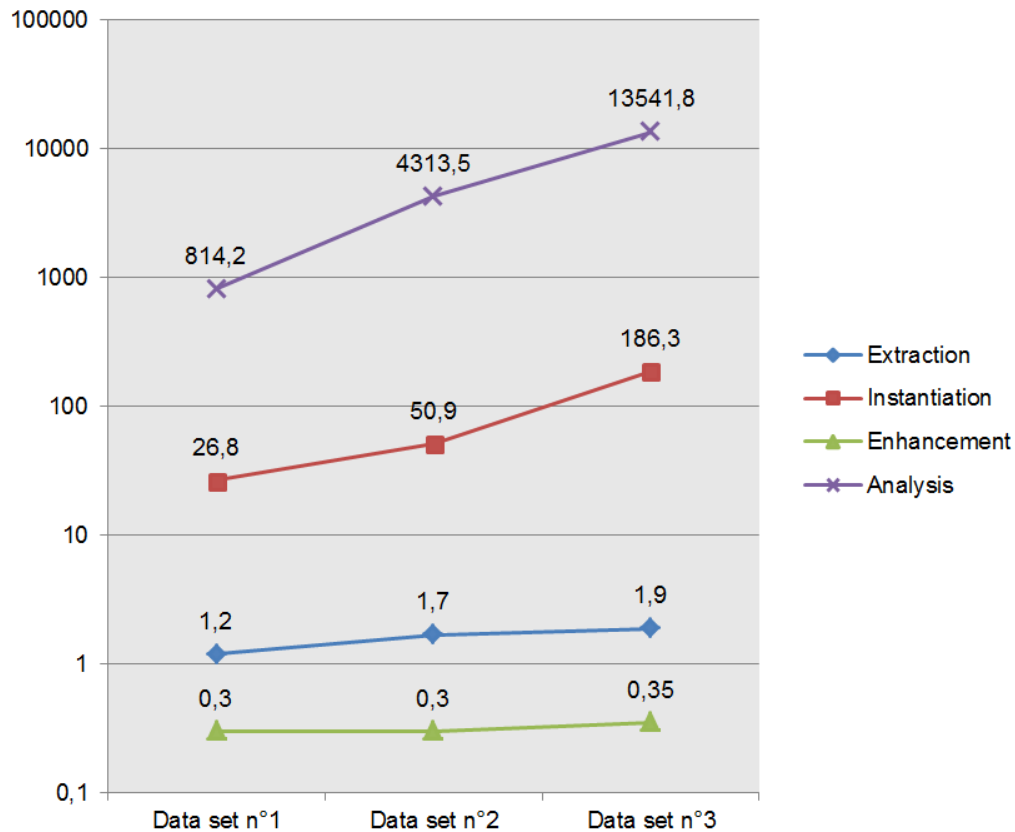


Figure 9.1: Execution times of the different steps of the process

9.3/ QUALITATIVE EVALUATION

The second task of the experiment illustrates the capabilities and the relevance of our approach. This study adopts the perspective of an investigator conducting an investigation using the tools presented in this thesis. Therefore, the investigator follows the investigation process PaLADIN, presented in Section 6.3.1 of Chapter 6, to carry out the investigation. To improve the clarity of this study, the steps of PaLADIN not directly related to the reconstruction of the events and their analysis will be omitted. It is assumed that, although not mentioned, these steps are carried out by the investigator.

The data set N°1 is used in the following case study. The investigation started after that

Phil, an employee of CompanyAndCo, has reported to his IT manager that a computer of the company generates abnormal and erratic behaviours. The computer restarts without the user requests it and the start sequence is very long (5 to 10 minutes). The computer is very slow and CPU usage increases and decreases without logical explanations. This computer is located in a common workspace and is used by several employees. Phil reported this abnormal behaviour after observing the previous symptoms during its last work session on this machine.

To understand what has happened, the investigator in charge started by collecting the hard disk of the suspect machine, and applied the SADFC approach to study the disk image. After successfully completing the extraction phase, the instantiation phase and the enhancement phase, the investigator starts the analysis by searching all significant correlations between events (*score* > 0.5). From the testimony of Phil on the behaviour of the machine, the investigator assumes that a malware is the cause of the problem. Therefore, he decided to search all traces of potentially suspicious executable that were run on the machine. To do this, he used the SPARQL query shown in Listing 9.1 to retrieve all executable that can be identified in the timeline. Among the results of the query given in Table 9.4, he identified two entries with a suspicious name (contagiomalware.exe) located in c:\users\andrew\downloads\ and c:\users\bobby\downloads\.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX ord2i: <http://checksem.u-bourgogne.fr/ord2i#>
SELECT ?path ?filename
WHERE
{
  ?exe a ord2i:WinExeFile.
  ?exe ord2i:hasLocation ?location.
  ?location ord2i:hasFilename ?filename.
  ?location ord2i:hasPath ?path
}
```

Listing 9.1: SPARQL query used to retrieve all the executables

The investigator then tried to understand who have interacted with this malware and in which circumstances. This information can be obtained using the correlation graph viewer. Indeed, the viewer allows to manually retrieve the events interacting with the executable files and then get information about the events. The information can also be retrieved using the SPARQL query given in Listing 9.2. This query allows to retrieve each event interacting with the executable named contagiomalware.exe. For each of these events, the query retrieves the name of the user involved, the type and subtype of the event, the date and time it occurred. The result of this query is shown in Table 9.5. The investigator noted that the users Andrew and Bobby have both used contagiomalware.exe. Andrew has downloaded it and then removed it. Bobby, on his side, has downloaded the malware, launched it and finally deleted it.

path	filename
c:\windows\temp\cr_e7cf0.tmp\	setup.exe
c:\windows\system32\	sppsvc.exe
c:\windows\system32\wbem\	wmiprvse.exe
c:\windows\system32\wbem\	wmiadap.exe
c:\users\andrew\downloads\	contagiomalware.exe
c:\programfiles\google\update\	googleupdate.exe
c:\windows\system32\	wermgr.exe
c:\windows\system32\	notepad.exe
c:\windows\system32\	smss.exe
c:\windows\system32\	csrss.exe
c:\windows\system32\	winlogon.exe
c:\windows\system32\	taskhost.exe
c:\windows\system32\	dllhost.exe
c:\windows\system32\	userinit.exe
c:\windows\system32\	dwm.exe
c:\windows\system32\	taskeng.exe
c:\windows\	explorer.exe
c:\programfiles\vmware\vmwaretools\	tpautoconnect.exe
c:\windows\system32\	conhost.exe
c:\programfiles\vmware\vmwaretools\	vmtoolsd.exe
c:\windows\system32\	searchprotocolhost.exe
c:\windows\system32\	searchfilterhost.exe
c:\users\bobby\downloads\	contagiomalware.exe
c:\programfiles\google\chrome\application\	chrome.exe
c:\windows\system32\	logonui.exe

Table 9.4: Results of the query retrieving all the executables

```

PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX time: <http://www.w3.org/2006/time#>
PREFIX ord2i: <http://checksem.u-bourgogne.fr/ord2i#>
SELECT ?pseudo ?type ?subtype ?datetime ?desc
WHERE
{
  ?user a ord2i:Person .
  ?user ord2i:hasPseudo ?pseudo .
  ?user ord2i:isInvolved ?event .
  ?event ?property ?exe .
  ?event ord2i:hasEventType ?type .
  ?event ord2i:hasEventSubtype ?subtype .
  ?event ord2i:hasDateTime ?date .
  ?event ord2i:hasDescription ?desc .
  ?date time:hasBeginning ?begin .
  ?begin time:inXSDDateTime ?datetime .
  ?property rdfs:subPropertyOf* ord2i:interacts .
  ?exe a ord2i:WinExeFile .
  ?exe ord2i:hasLocation ?location .
  ?location ord2i:hasFilename ?filename .
  FILTER( ?filename="contagiomalware.exe" )
}

```

Listing 9.2: SPARQL query retrieving information about events interacting with contagiomalware.exe

The investigator then tried to find out if both Andrew and Bobby are related and if so, why? To answer these questions, he uses the correlation graph viewer. The consultation of a

pseudo	type	subtype	datetime
Andrew	Chrome History	File Download	2016-01-03T07:21:16.000+01:00
Bobby	Chrome History	File Download	2016-01-03T08:12:09.000+01:00
Bobby	Recycle Bin	File Deletion	2016-01-03T08:12:43.000+01:00
Bobby	Windows Prefetch	App. Prefetch	2016-01-03T08:12:25.000+01:00

Table 9.5: Results of the query retrieving information about events related to contagioma-
lware.exe

wide view of the graph allows to distinguish different clusters of events (see Figure 9.2). In particular, there are three clusters corresponding to the events belonging to each user. A quick study of the cluster related to Phil shows that he browsed the web and consulted his emails. The most relevant information for the investigation is located in the clusters related to Andrew and Bobby. The existence of many links between these two clusters indicates that these two users have strong interactions. The investigator then zooms on the small group of events between the two clusters; looking for information about the use of the executable contagiomalware.exe by one of the two users. After having found one of the events, he can view the closest neighbourhood of the event using correlation links shown in Figure 9.3. The chain of correlated events found represents the life cycle of the malware on the computer. By clicking on the nodes, he can consult the information about the events (date and time, related objects and subjects, description).

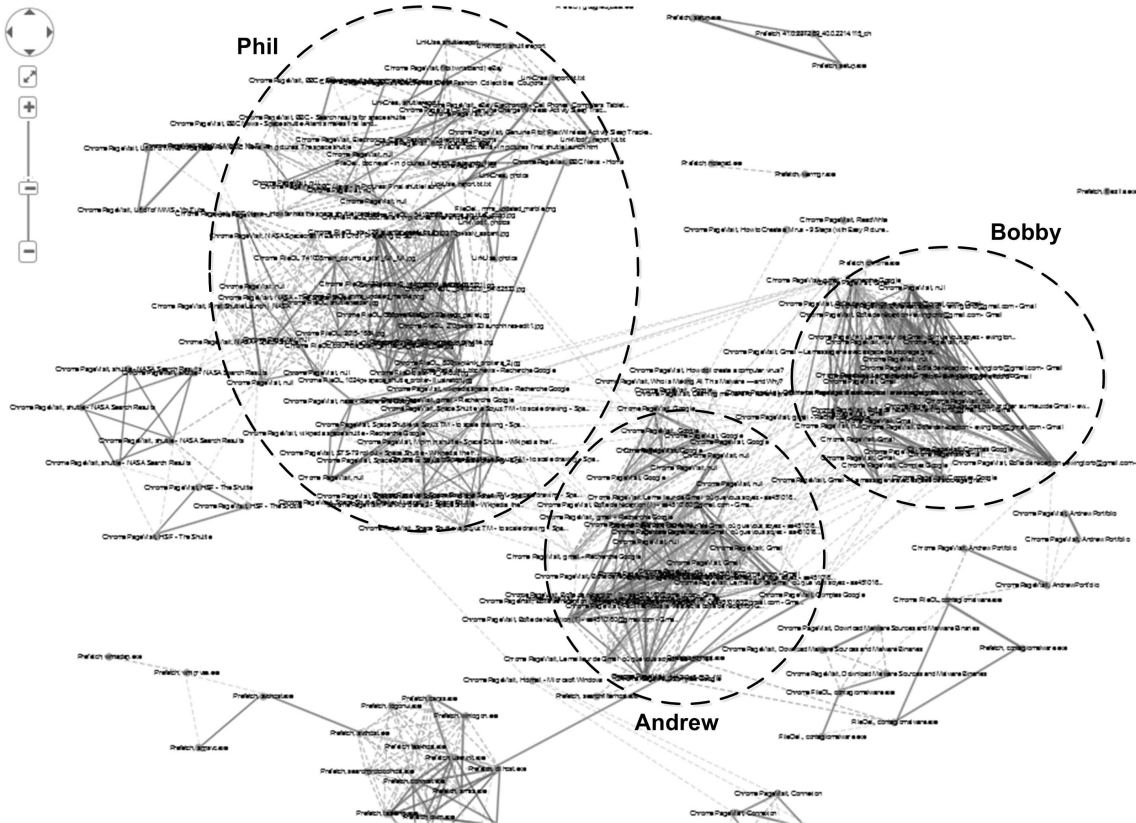


Figure 9.2: Correlation graph viewer: wide zoom on the events composing the case

To get more details about an event, he can also use the SPARQL interface. The use of the correlation graph viewer and the SPARQL interface, two complementary tools, allows him to identify the following information. Regarding Andrew, we can see that he has viewed a website entitled "Download Malware Sources and Malware Binaries" at <http://malwarewebsite.com>. He has then downloaded the file `contagiomalware.exe` from this website. He then went on his personal website named "Andrew Portfolio". By clicking on the corresponding node, we obtain the URL of Andrew's website <http://andrew-and-co.com>. Regarding Bobby, we can see that he visited the website named "Andrew Portfolio". He consulted on this website a folder called "private" from which he downloaded the file `contagiomalware.exe`. The graph also confirms that Bobby launched this executable and then deleted it.

In conclusion, it appears clearly that Andrew and Bobby have strong interactions. We see that there are numerous links between two of the three users of the machine: Bobby and Andrew. This means that they share a special connection and that they are potentially accomplices. The study of events chains in the viewer allows to understand the nature of their interactions. In particular, we see that they both used the file `contagiomalware.exe` and the website <http://andrew-and-co.com>. Current tools of SADFC do not allow to determine with certainty the nature of this interaction. However, the investigator may assume that Andrew actually downloaded the malware from the website <http://malwarewebsite.com> and he then made it available on his personal website <http://andrew-and-co.com>. Bobby has then downloaded and launched the executable. The effects on the machine remain unknown at this stage of the investigation. The viewer also shows that Phil did not interact much with the other two users and his activities can be summarised as Web browsing.

As shown in Section 8.5 of Chapter 8, the investigator can then get information and explanations to support the previous conclusions. Using the SPARQL endpoint to access the knowledge contained in the TKL layer of the ontology, he can, for example, get information about several relevant correlation scores (i.e. get the scores used to compute the overall correlation score), get explanations about the deductions made during the enhancement phase or determine how and when each piece of information was extracted from the crime scene.

9.4/ CONCLUSION

The evaluation of a digital forensics tool includes several criteria, covering both technical aspects and legal aspects. In this chapter, a complete evaluation of the tools proposed in the ANNALIST architecture (which implements the SADFC approach) was proposed.

This evaluation has first quantified the data volumes handled by the tools and the execution times of the various steps composing the reconstruction process. Indeed, from

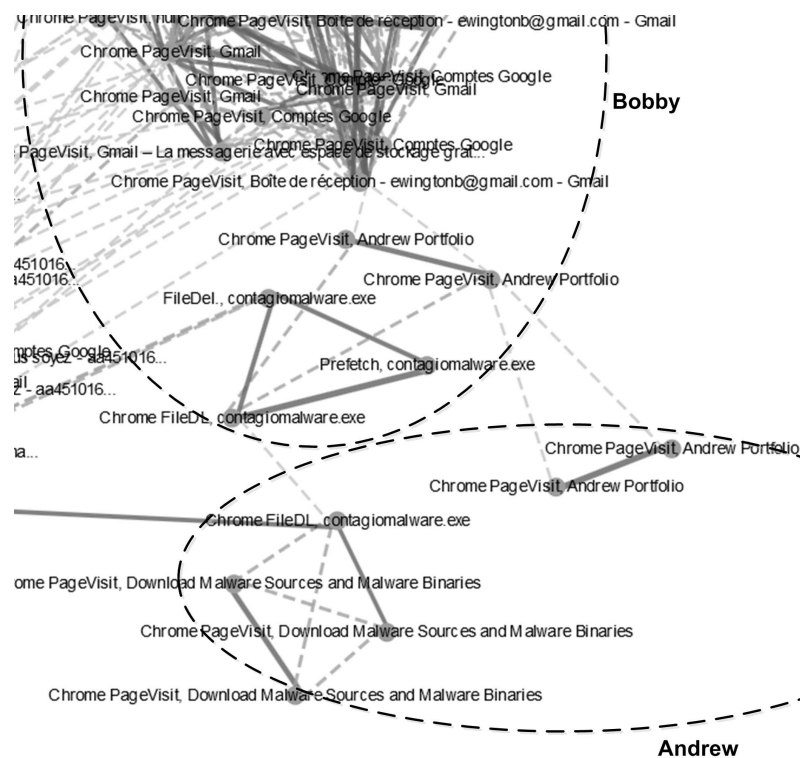


Figure 9.3: Correlation graph viewer: close view on the chain of events related to contagiomalware.exe

the point of view of performance, the proposed tools must be able to handle large volumes of data in a time compatible with the needs of justice. Regarding this criterion, the performance of the tool can still be improved, especially the execution time of the analysis phase. The benchmarks show that it takes approximately three hours to handle a timeline with 20000 entries which is a small timeline compared to the timeline handled in real cases. The optimisation of the analysis is, therefore, a priority in future versions of the tool. Regarding the extraction and the instantiation of the ontology, the tool can presently handle fifteen sources of information which is still not enough to have a complete view of an incident. In future work, new sources will therefore be integrated. In particular, information sources related to Android and iOS must be integrated to handle mobile devices.

In the second part of the experimentation, the relevance of the proposed tools was demonstrated through a fictitious scenario. First, we have seen that the proposed approach allows to quickly extract the knowledge related to a digital incident. This knowledge can then be enriched using the enhancement process which allows to deduce new knowledge from the existing one. The reconstruction of the timeline is fully automatic and simple inferences are used to rapidly enrich it. After the analysis phase, the use of the correlation graph viewer and the SPARQL interface allow to understand the nature of the interactions between the users composing the case. However, the proposed tool pro-

vides research directions, but the analytical faculties of the investigators are still required to complete the investigation. For example, in the proposed case study, the investigator brought the initial idea (i.e. the assumption that the erratic behaviour is caused by a malicious program). From this starting seed, the tools quickly enable him to answer questions like "What is the malicious program involved?", "Who executed this program?" and "How this program has arrived on the machine?". In particular, the identification of correlations between events allows to build chains of related events that are thereafter navigable via the correlations viewer. Once the suspicious program is identified, the correlation tool can identify the entire life cycle of it and the users who interact with this resource. In conclusion, this case has shown that the analysis associated with the visualisation tools help to increase the effectiveness of the investigators. In addition, the knowledge contained in the TKL layer of the ontology can help the investigators to explain to Justice the results produced by the tools.

CONCLUSION

CONCLUSION

The event reconstruction is a process composed of two phases: the specification of events and the analysis of the timeline containing these events. The first phase aims to collect the footprints left by the activities of the protagonists of an incident by studying the digital objects found in the crime scene. The information contained in these footprints is used to determine the events that occurred in the past, in order to build a representative timeline of the past situation. The second phase comes after the construction of the timeline. It involves the analysis of the timeline to understand the situation, to determine the nature and the circumstances of the incident and finally, to establish the role of each protagonist. This process covers many tasks including the identification of illicit events and the identification of correlations between events.

The reconstruction and the understanding of an incident is a complex cognitive process that requires on the one hand to restore the original meaning of each footprint collected in the crime scene, and on the other hand to build a picture of the incident that brings up the semantic relationships between these semantic entities. This intellectual work is made difficult by the large volumes of data that need to be handled to build the timeline. The large amount of information to federate within the timeline and subsequently the analysis of this timeline are particularly laborious and complex tasks for the investigators. They face significant cognitive overload that can cause them to make mistakes or miss important information. Moreover, the dispersion of information across multiple digital devices and the syntactic and semantic heterogeneity of data make it difficult to build a timeline representing accurately and completely what happened during the incident. Finally, the reconstruction of events, as any forensic activity, is governed by legal requirements. In particular, the results produced at the end of the reconstruction must meet a high level of credibility and reproducibility to be admissible in a court.

10.1/ CONTRIBUTIONS OF THE THESIS

To address these problems, this thesis sets seven goals which are important steps towards the resolution of the research problems addressed in this thesis. The definition of these goals is a contribution resulting from a thorough study of the field of computer forensics. These goals are recalled briefly below:

- Provide tools to **generate automatically the timeline** from all the information that can be found in a crime scene.
- **Assist the investigators in the analysis of the timeline.**
- **Facilitate the understanding and the manipulation of the knowledge** collected during the investigation by making available ergonomic and intuitive visualisation tools.
- Propose a storage model which **federates all the knowledge collected from the crime scene.**
- **Solve the technical problems related to the heterogeneity of data.**
- Ensure the **credibility of the results** produced by the tools.
- Ensure the **reproducibility of the methods** of investigation.

To reach these objectives, the SADFC approach is proposed. SADFC is a synergy of elements defining the methods and the tools needed to complete the reconstruction of events.

To satisfy the legal requirements inherent to the field, SADFC is first based on theoretical foundations. It introduces formal definitions of a crime scene and the entities composing it in order to disambiguate these concepts. Based on this formalisation, operators are proposed to carry out the various tasks composing the reconstruction of events, from the extraction of the footprints to the analysis of the timeline. The formal nature of these operators helps satisfy the need for credibility by making their operation explicit. These operators have been presented in an international peer-reviewed journal (Chabot et al., 2014b). To ensure the interface of event reconstruction operators in the broader process of the investigation, they are integrated into an investigative process named PaLADIN. This process model describes all the steps composing the investigation and thus meets the need for reproducibility via an explicit description of the investigation process. In addition, the ontology proposed in our work incorporates a layer dedicated to the representation of activities during the investigation. The storage of the nature of each task in addition to the information used as input and output of them helps understand how each result is produced. This feature can be used by the investigators to support their results with clear explanations.

The principles and tools developed in the theoretical foundations of SADFC are implemented as an architecture named ANNALIST which has been presented in an international peer-reviewed journal (Chabot et al., 2015b) and three international conferences

(Chabot et al., 2015a, 2014d,a). This architecture provides users the functionality needed to complete both phases of the reconstruction of events and is designed to provide answers to five of the seven goals pursued by this thesis. The proposed architecture is divided into four layers taking advantage of an ontology named ORD2I.

The **extraction layer** and the **knowledge layer** associated to the ontology offer the ability to generate automatically timelines from the footprints left on a hard disk. The extraction layer, thanks to the use of the Plaso toolbox, is able to extract large amounts of heterogeneous footprints contained in a wide range of sources of information in a short amount of time, as it has been demonstrated by the benchmarks in Chapter 9. This information is then used by the knowledge layer to instantiate the central element of the architecture, the ORD2I ontology. This ontology federates the knowledge collected from the crime scene and enables the representation of a digital incident and the events composing it. To enable the analysis tools to produce informed and relevant conclusions, ORD2I offers an accurate and faithful representation of the crime scene integrating within it information from numerous sources. This is made possible by the expressiveness of OWL 2 and by the integration of several knowledge layers including both common and specialised information about the entities composing the crime scene. This separation into two layers enables the analysis tools to reason on the generic level (time, objects interacting with the events, subjects involved in the events) and/or on the technical information about objects.

To assist the investigators in the analysis of the timelines and to reduce the effects related to cognitive overload, we have implemented an automated analysis tool based on an ontology to facilitate the understanding of the information and to emulate cognitive processes used by the investigators. Thanks to its formal nature and its ability to structure the knowledge, the ontology facilitates the understanding of the information by the analysis processes. Ontological languages like OWL 2 are able to strongly structure the information by representing it in the form of concepts, properties and logical constraints. This language thus enables the building of a structured representation of events giving analytical tools some understanding of the information extracted from the crime scene. At the end of the instantiation process, a knowledge graph incorporating the information contained in the textual output of Plaso is obtained. This graph is then used by the **reasoning layer** to provide users with advanced analytical tools and intuitive knowledge visualisation/manipulation interfaces. Regarding the analysis part, we present in this thesis how the inference mechanism can perform simple deductions instead of the investigators and thereby enriching the timeline automatically. A tool to identify correlations between events is also presented. This takes advantage of generic knowledge (time, objects and subjects) and technical knowledge (EK-based rules) contained in the ontology to detect events that are potentially correlated. The identification of correlations, as shown in Chapter 9, allows to build chains of events. These chains can be browsed by the users to know the causes, the effects or the context of a given event.

Finally, the **interface layer** is intended to facilitate the understanding and the manipulation of the knowledge. This layer provides a correlation viewer that presents information as graphs whose nodes are the events constituting the timeline and links are the correlations between events. As it is shown, this viewer allows the investigators to have an overview of the case and quickly understand the physiognomy of the incident. In addition, the user can consult all the knowledge contained in the base to get additional information on an event, resource, or the activities of a person. This is made possible through the power and expressiveness of the SPARQL query language. In addition to the functions provided by the interface layer, the work of the investigators is simplified by the inherent qualities of the ontology.

10.2/ ISSUES AND FUTURE WORK

To complete the work of this thesis and enhance the quality of the tools developed, several axes of research and several suggestions for possible improvements are considered.

First, on the formal reconstruction operators, an operator to identify illicit activities is required to complete the theoretical foundations of SADFC as it is one of the steps of the process model PaLADIN. Currently, this task must be performed manually by the investigators. The identification of illicit actions is a complex research problem addressed in many works, especially in the field of intrusion detection systems (Debar et al., 1992), (Ilgun et al., 1995), (Depren et al., 2005), (Karthick et al., 2012). An illicit action may be composed of one or more events. Indeed, an illegal act can sometimes be a serie of events which, if taken separately, are considered legal, but which taken together form a wrongdoing. To identify the illicit actions, one approach is to use pattern-based detection tools. However, as said in Chapter 3, this type of tools suffers from several limitations, including the difficulty to build an exhaustive set of signatures and the time required to keep it up to date. A possible alternative is the use of machine learning algorithms trained using a base of illicit actions manually identified.

PaLADIN also has several limitations that will be the subject of future works. The first limit concerns the assessment of the relevance of this model. PaLADIN is the result of a synthesis of a state of the art and it has not been assessed or used in real situations by forensic experts for the moment. PaLADIN has not been tested on a panel of cases sufficiently diversified to be able to argue that this model has a good adaptability to all types of cases. A second limitation is the accuracy of the model that can be further enhanced. Indeed, PaLADIN defines the concepts of footprints and events and give details about operations performed by each task of the event reconstruction process on these entities. However, the level of detail can be increased by specifying more information about entities (e.g. the nature of the events, the type of information sources used, etc.). Moreover, although PaLADIN integrates these notions in the event reconstruction phases,

these concepts are lacking in the rest of the digital phases of the investigation and from the entire physical investigation. It is therefore necessary to homogenise the process model by using a similar level of precision in the other phases of the investigation.

Regarding the ORD2I ontology, it was subject to peer review through the publication of an international journal paper (Chabot et al., 2015b). However, this ontology has not been used and validated on real cases by forensic scientists. Nevertheless, as part of this research project, we have recently joined a working group with the goal of adopting a knowledge representation for the event reconstruction. This working group is composed of representatives of companies and research teams specialised in computer forensics. This initiative allows us to present ORD2I to the experts of the field in order to improve its quality.

Concerning the architecture, the performance can still be improved, especially the execution time of the analysis phase. The benchmarks show that it takes approximately three hours to handle a timeline with twenty thousand entries which is a small timeline compared to the timeline handled in real cases. The optimisation of the analysis is, therefore, a priority in future versions of the tool. One solution for improving performance is to reduce the quantities of data to be processed. For this, a filtering function is already implemented in the architecture. It allows the users to filter some information sources by selecting them in the settings of the application. This filtering system must be refined to enable users to manage data more precisely. Indeed, for some sources of information, the user may wish to filter only some types of events and thus, do not completely filter out the source but only part of the information contained in it.

The extraction and instantiation tools can presently handle fifteen sources of information which is still not enough to have a complete view of an incident. In future work, new sources will therefore be integrated. In particular, information sources related to Android and iOS must be integrated to handle mobile devices. At the same time, the ontology ORD2I must be validated by experts.

Several visualisation tools will be implemented in future works. First, more features will be integrated in the correlation graph viewer including search functions. Second, an advanced timeline viewer will be proposed. This viewer will allow on the one hand to display conventionally the timeline and on the other hand to enrich this timeline with knowledge from the ontology including information about objects and subjects interacting with each event and information produced by the enhancement phase and the analysis phase. There are numerous works on visualisation of traces (Cram et al., 2007) that can be used as reference to determine the relevant features (filtering, sorting, grouping, etc.) to provide in our tools.

We also plan to allow the investigators to enrich the knowledge of the ontology with information they found on their own on the digital crime scene in order to improve interactivity and collaboration between SADFC and the investigators. The implementation of

such functionality requires mechanisms to check the consistency of the ontology. Indeed, adding new information may conflict with the existing knowledge about the incident or enables new deductions. Furthermore, if several investigators work on the same case, we must provide collaborative tools to manage the points of view of all members of the team.

APPENDICES

SEMANTIC WEB

The semantic web technologies are used in this thesis to address issues in the field of digital forensics. This appendix is intended to give a quick overview of the field of the semantic web to the readers by presenting succinctly the techniques and the technologies used in our work. This appendix is structured in the following way. Section A.1 and Section A.2 introduces respectively the field of semantic web and the stack of technologies used in this field. Section A.3 then introduces the notions of knowledge base and ontology which are the backbone of the semantic web. Section A.4 then presents the languages used to define ontologies, including RDF, RDFS and OWL. Finally, Section A.5 briefly presents how a knowledge base can be exploited using SPARQL and reasoning. This chapter is based on a technical report (Chabot, 2012). For reasons of brevity, this appendix succinctly introduced technologies. This appendix aims to give an overview of the tools used in our work.

A.1/ A BRIEF INTRODUCTION TO THE SEMANTIC WEB

The evolution of new technologies and their use has resulted in a rapid increase of the quantity of data produced each day. The democratisation of services such as YouTube, Facebook, Flickr, etc. has resulted in the production of huge amounts of data every day. It is therefore increasingly difficult for the users to manipulate and process these large volumes of data. Created by Tim Berners Lee, the semantic web is a set of technologies whose main goal is to make available to machines the meaning of data. The semantic web was created to use the increasingly high processing capacity of machines to process the volumes of data on the Web by providing the means to the machines to process this data efficiently and correctly instead of humans. This enables to assist the users in the process of the data. The semantic web can be considered as a sub domain of knowledge engineering as it brings elements of response to several problems inherent in this area.

- The representation of knowledge in sectors where the need to collect and manage large volumes of knowledge is omnipresent.

- Search for information through advanced search engines. In this area, the use of semantics allows the engines to give more relevant answers to the users.
- The sales websites where the semantic web can meet the needs inherent to user profiling and recommendation of items based on these profiles.

In our work, the semantic web technologies are used in a non-web context to address problems encountered in the field of computer forensics. However, the philosophy of the semantic web is preserved in our application as it consists to give to analysis processes some understanding of the data to process, in order to provide advanced functionalities to assist the users.

A.2/ SEMANTIC WEB ARCHITECTURE

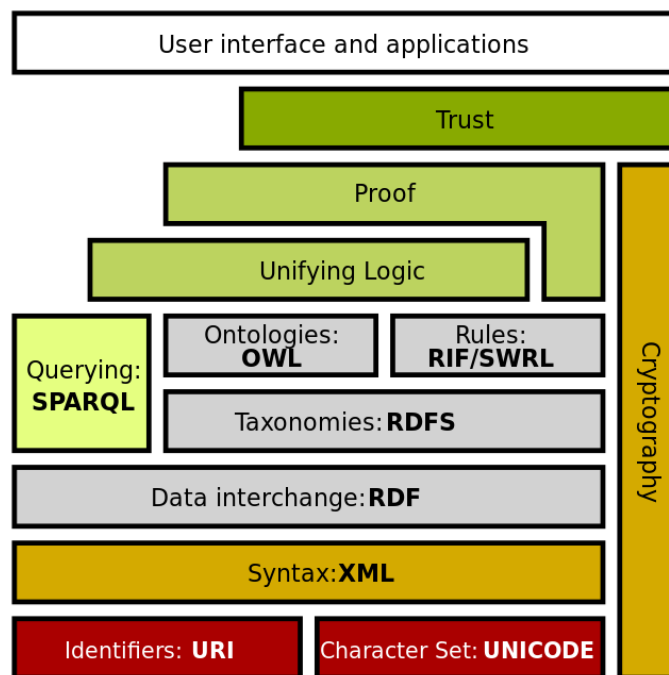


Figure A.1: Semantic web stack

The semantic web is a set of technologies organised in the form of a stack, as illustrated in Figure A.1:

- The "Identifiers" layer aims at identifying resources through the use of URIs.
- The "Syntax" layer provides a meta-language for representing information in a structured way. This layer contains, in addition to XML, several other technologies including DTD (Document Type Definition), XML Schema and the formatting language XSL.
- The "Data Interchange" layer is the first element of the architecture, providing means to represent the semantics of the data. RDF is a language enabling to describe

resources through the use of triples ($\langle \text{subject}, \text{predicate}, \text{object} \rangle$).

- The "Taxonomies and Ontologies" layers are intended to extend the expressiveness of RDF.
- The "Logic and Proof" layers respectively allow to use inference mechanisms and to prove the truthfulness of the knowledge produced by them.
- The "Trust" layer is intended to explain the reasoning produced by the previous layer and ensure its reliability.

It should be noted that an application can use only a subset of this stack, depending on its needs. Building an application taking advantage of the semantic web technologies almost systematically involves the creation of a knowledge base, a receptacle containing the knowledge used in the application. To enable machines to understand the meaning of such knowledge, the knowledge base must provide a structured and formal representation of the data. To reach this objective, the knowledge base introduced in our work is structured using an ontology. The notions of knowledge base and ontology are presented in the following section.

A.3/ KNOWLEDGE BASE AND ONTOLOGY

A knowledge base is a structure providing, on the one hand, means to represent facts, and, on the other hand, tools to reason on this knowledge. It is composed of two elements: a schema and individuals instantiating this schema (Hepp, 2008). The schema is called TBox (i.e. Terminology Box) and the individuals and all the assertions related to them are called the ABox (i.e. Assertional Box). The TBox defines the organisation of the knowledge using concepts, properties and constraints on both of them. The TBox of a knowledge can be formalised using the Karlsruhe Ontology Model (Ehrig et al., 2005):

$\nu := (C, \leq_C, R, \sigma_R, \leq_R, A, \sigma_A, T)$ Where

- C, R, A and T are disjoint sets containing respectively the concepts, the properties, the attributes and the datatypes (integer, string, etc.) composing the TBox.
- \leq_C is a partial order on concepts organising them into a hierarchy. This hierarchy is the taxonomy of the knowledge base.
- σ_R is a function associating concepts to each property using the notions of domain and range. The domain of a property represents the subject of a property while the range represents the object.
- \leq_R is a partial order on properties organising them into a hierarchy.
- σ_A is a function associating the concepts, the attributes and the datatypes.

The following example illustrates the definition of a TBox:

- $C = \{\text{LivingCreature}, \text{Animal}, \text{Cat}, \text{Dog}, \text{Human}, \text{Man}, \text{Woman}\}$
- $\leq_C := (\text{Animal} \leq \text{LivingCreature}, \text{Human} \leq \text{LivingCreature}, \text{Cat} \leq \text{Animal}, \text{Dog} \leq$

- $Animal, , Man \leq Human, , Woman \leq Human)$
- $R = \{hasPet, hasHusband\}$
- $\sigma_R(hasPet) = (Human, Animal)$
- $\sigma_R(hasHusband) = (Woman, Man)$
- $\leq_R := \emptyset$
- $A = \{hasName, hasWeight, hasAge\}$
- $\sigma_A(hasName) = (LivingCreature, string)$
- $\sigma_A(hasWeight) = (LivingCreature, float)$
- $\sigma_A(hasAge) = (LivingCreature, integer)$

The previous definition has formalised the structure of the knowledge base. The population of the knowledge base ν (i.e the ABox) can be represented as follows: $Population := (I, i_C, i_R, i_A)$

Where

- I is a set representing the individuals.
- i_C is a function associating individuals to concepts.
- i_R is a function associating individuals to properties.
- i_A is a function associating giving values to the attributes of the individuals.

The following definitions illustrate the definition of the ABox:

- $i_C(Cat) = \{garfield, azrael\}$
- $i_C(Dog) = \{santaLittleHelper\}$
- $i_C(Man) = \{homer, gargamel, jon\}$
- $i_C(Woman) = \{liz, marge\}$
- $i_R(hasPet) = \{(jon, garfield), (gargamel, azrael), (homer, santaLittleHelper)\}$
- $i_R(hasHusband) = \{(marge, homer)\}$
- $i_A(hasName) = \{(homer, "HomerSimpson"), (liz, "LizWilson"), (jon, "JonathanArbuckle"), (gargamel, "Gargamel")\}$
- $i_A(hasWeight) = \{(garfield, 15)\}$
- $i_A(hasAge) = \{(gargamel, 93), (homer, 45)\}$

In most of the cases, the schema used to structure the knowledge base is an ontology. This ontology is then instantiated, which means that individuals are created and assertions are made on these individuals. Ontology is a central concept of the semantic web that was defined as an explicit specification of a conceptualisation using a declarative formalism by (Gruber et al., 1993). The explicit and formal nature of the ontology allows the construction of a knowledge representation comprehensible by the machines. Thus, using ontology languages, machines are able to understand the meaning of data without prior knowledge about the domain and advanced reasoning mechanisms can be build on top of this representation.

A.4/ ONTOLOGY LANGUAGES

This section introduces the three main languages used to formalize knowledge bases. After explaining how each resource is identified, the languages RDF, RDFS and OWL are presented.

A.4.1/ URI AND NAMESPACES

URI (Uniform Resource Identifier) is a global naming system allowing to designate any physical or abstract resource without unambiguously. Given the unique name assumption, several URIs can reference the same resource. However, a single URI can not reference several resources. http://dbpedia.org/page/Sherlock_Holmes and <http://dbpedia.org/page/London> are two examples of URI. URI combines two elements. First, the address of the resource is given using a URL (Uniform Resource Locator). This locator specifies where the resource is located. The URL may be subject to changes during the life of the resource. Second, the identity (or name) of the resource is given using a URN (Uniform Resource Name). A URN is a persistent identifier of the resource.

An XML namespace is a collection of terms which are used in other documents as element types and attribute names. This namespace is designated by a URI. The use of namespaces in a document makes it possible to use multiple vocabularies defined by other people. This mechanism is used extensively in RDF, RDFS and OWL. In this manuscript, several namespaces are used for the definition of the ORD2I ontology:

- xsd (<http://www.w3.org/2001/XMLSchema#>) is a namespace mainly used for datatypes.
- rdf (<http://www.w3.org/1999/02/22-rdf-syntax-ns#>) contains the terms inherent to RDF.
- rdfs (<http://www.w3.org/2000/01/rdf-schema#>) contains the terms inherent to RDFS.
- owl (<http://www.w3.org/2002/07/owl#>) contains the terms inherent to RDFS.

A.4.2/ RDF/RDFS

RDF is a W3C recommendation based on the notion of graphs to describe resources. RDF documents are structured in the form of triple <subject, predicate, object> where the subject is the resource described, the predicate is the type of property used and the object is the property value for the subject. A triple represents a fact, such as "Gargamel is 93 years old" and "Jon has a pet named Garfield". Each of these facts is composed of a subject (Gargamel, Jon), a predicate (has age, has a pet) and an object (93, Garfield). In RDF, the subject and the predicate are designated by a URI when the object can be

designated by a URI if it is an entity or by a literal if the object is a value (a string, an integer, etc.). The RDF Schema language was created to bring better structure to RDF documents. RDFS provides the following elements (this list is not exhaustive):

- *rdfs:Class* is used to declare a resource as a class.
- *rdfs:subClassOf* provides the ability to define classes hierarchy.
- *rdfs:domain* defines the type of resources used as the subject of a given property.
- *rdfs:range* defines the type of resources used as the object of a given property.

Figure A.2 is a RDF representation in a graphical form of a portion of the facts shown in the knowledge base of Section A.3.

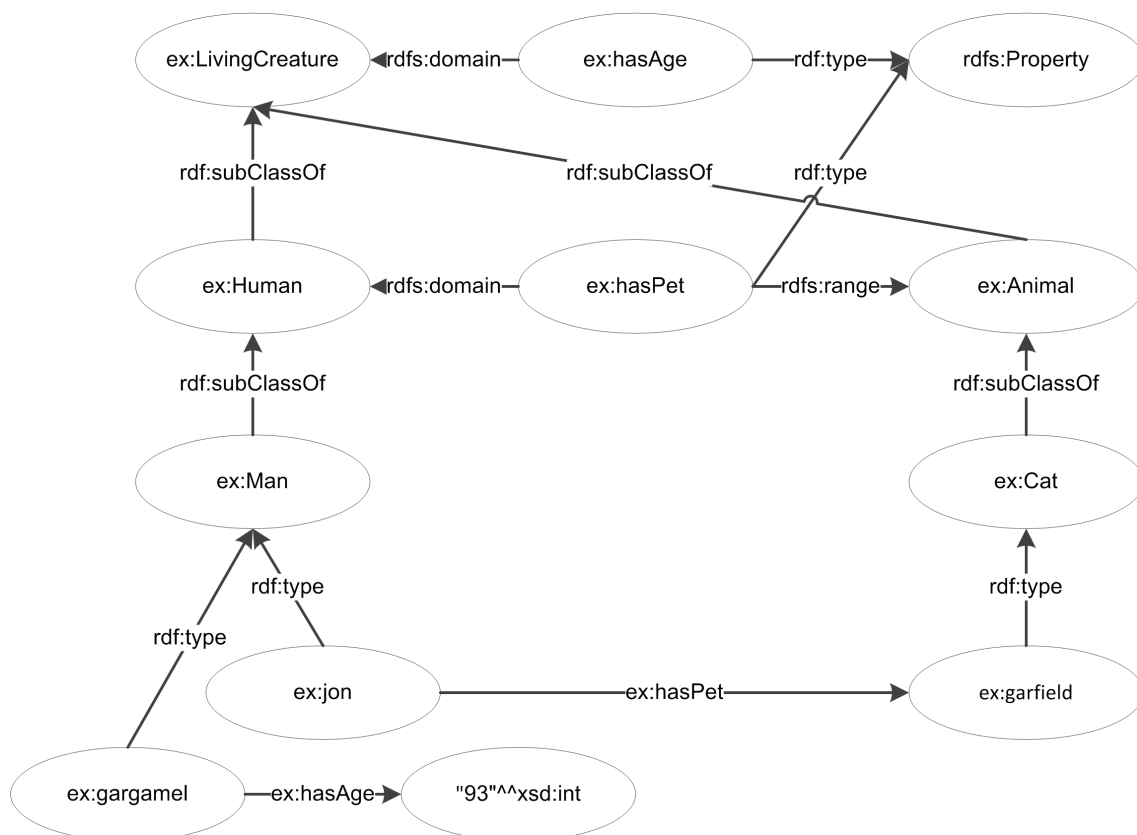


Figure A.2: Representation of facts using RDF/RDFS

RDF/RDFS (as well as OWL) can be serialised in various formats. In this manuscript, the Turtle serialisation is used. Listing A.1 is the serialisation of the graph presented in Figure A.2.

```

@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix ex: <http://exemple.com/> .
ex:LivingCreature rdf:type rdfs:Class .
ex:Human rdf:type rdfs:Class ;
  rdfs:subClassOf ex:LivingCreature .
ex:Man rdf:type rdfs:Class ;

```

```

    rdfs:subClassOf    ex:Human .
ex:Animal    rdf:type    rdfs:Class ;
    rdfs:subClassOf    ex:LivingCreature .
ex:Cat    rdf:type    rdfs:Class ;
    rdfs:subClassOf    ex:Animal .
ex:garfield    rdf:type    ex:Cat .
ex:jon    rdf:type    ex:Man ;
    ex:hasPet    ex:garfield .
ex:gargamel    rdf:type    ex:Man ;
    ex:hasAge    "93"^^xsd:int .

```

Listing A.1: Representation of facts using RDF/RDFS and Turtle

A.4.3/ OWL

OWL (Ontology Web Language) is a W3C recommendation used to define and instantiate ontologies. This language is an extension of RDF and RDFS. OWL is designed to fill a lack of expressiveness present in these two languages. In particular, it introduces the notions of equivalent classes and equivalent properties, the equality between individuals, the symmetric properties or the restrictions on values. OWL includes three sub-languages:

- OWL Lite is the least expressive of the three proposed languages. It is particularly suitable for applications that needs a greater expressiveness than RDF/RDFS while maintaining the simplicity of use.
- OWL DL offers a more expressive language than OWL Lite. All the properties of OWL are present in this language. However, a set of constraints has been set to ensure that every reasoning task can be solved (i.e. completeness) in a finite time (i.e. decidability).
- OWL Full is a sub-language with a maximum expressiveness. However, this expressiveness leads to the impossibility to guarantee that the reasoning tasks can be solved in a finite time.

In this thesis, the second version of OWL is used. OWL 2 is a W3C recommendation from 2009 based on OWL DL. OWL DL is based on description logics and thus benefits from the properties of the latter including formal semantics, completeness and decidability. OWL 2 offers multiple profiles to suit the user's needs. The ontology presented in this manuscript is implemented using OWL 2 RL (Rule Language), a profile restricting the OWL language to improve the response times of queries and reasoning tasks. OWL 2 RL enables the definition of class hierarchies and property hierarchies using the terms *owl:Class*, *rdfs:subClassOf* and *rdfs:subPropertyOf*. OWL distinguishes two types of properties. The object properties (*owl:ObjectProperty*) are predicates connecting two classes together while datatype properties (*owl:DatatypeProperty*) links a class with a datatype. Like in RDFS, the terms *rdfs:domain* and *rdfs:range* are used to define the subject and the object of a property. OWL 2 RL also allows to define additional fea-

tures to properties. The terms *owl:transitiveProperty* and *owl:symmetricProperty* allow respectively to declare a transitive property and a symmetric property. If p is a transitive property, then if $\langle ?a, ?p, ?b \rangle$ and $\langle ?b, p, ?c \rangle$ are two facts of the knowledge base, then the fact $\langle ?a, p, ?c \rangle$ can be deduced. If the property p is a symmetric property implies that if the fact $\langle ?a, p, ?b \rangle$ is declared, then the fact $\langle ?b, p, ?a \rangle$ can be deduced. Finally, OWL offers the means to instantiate the classes and add features on these individuals.

A.5/ MANIPULATION OF ONTOLOGIES AND REASONING

A.5.1/ SPARQL

SPARQL is a query language for accessing and manipulating triples contained in a knowledge base. Like the SQL language, SPARQL allows to explore the data using complex queries made of selection, filters, aggregation and other functions. A Select query is composed of two keywords: *SELECT* and *WHERE*. For example, Listing A.2 gives an example of query selecting the name and the URI of all people with age above 50. For this, the query engine searches the facts contained in the knowledge base that can match with the facts given in the *WHERE* clause. In our case, this query returns *ex:gargamel* and "Gargamel".

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX ex: <http://exemple.com/>
SELECT ?human ?name
WHERE
{
  ?human rdf:type ex:Human.
  ?human ex:hasName ?name.
  ?human ex:hasAge ?age.
  FILTER(?age > 50).
}
```

Listing A.2: Example of selection using SPARQL

The applications presented in this thesis also requires the updating of knowledge contained in the base. For this, the SPARQL Update (SPARUL) language is used. This language offers the possibility to insert, modify and delete triples using the keywords *INSERT DATA* and *DELETE*. The queries given in Listing A.3 and Listing A.4 allow respectively to insert and delete facts. The first query inserts two new facts in the base while the second query deletes all records of old creatures (age greater than 50).

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
```

```

PREFIX ex: <http://exemple.com/>
INSERT DATA
{
    ex:azrael    rdf:type    ex:Cat .
    ex:gargamel  ex:hasPet    ex:azrael .
}

```

Listing A.3: Example of insertion of new facts using SPARUL

```

PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX ex: <http://exemple.com/>
DELETE
{
    ?creature    ?p    ?o
}
WHERE
{
    ?creature    ex:hasAge    ?age .
    FILTER(?age > 50)
}

```

Listing A.4: Example of deletion of facts using SPARUL

A.5.2/ REASONING

As mentioned above, the purpose of the semantic web is to make available to the machines the significance of the data to enable them to reason on it. This section gives a quick overview of reasoning possibilities offered by the semantic web technologies.

A first type of reasoning is the inference reasoning. The inference is an operation consisting in drawing a conclusion from known knowledge. This makes it possible to deduce new information from the existing information. It is a creative process that allows the expansion of the knowledge base. To carry out the inference tasks, reasoners and rule engines can be used. In our work, we choose to make inference using SPARQL. Select queries are used to search in the knowledge base facts that corresponds to a given inference case. When such facts are discovered, the conclusion (new facts) is added to the base using the SPARUL language.

A second application of reasoning is the verification of the coherence and consistency of the ontology. A knowledge base is incoherent when its TBox is. A TBox is declared inconsistent when one of its concepts is unsatisfiable. A concept is said unsatisfiable if for every model of ontology, the number of instances of the concept is zero (a model is an interpretation that satisfies all axioms of ontology). The inconsistency, on its side, is a characteristic of the relation between the ABox and the TBox. An ABox is declared consistent for a given TBox if there is an interpretation that is a model for the TBox and

the ABox. If there is no model, the inconsistency is then declared. This type of reasoning makes it possible to verify that the facts contained in the knowledge base follow the rules established in the ontology. The verification of the coherence and consistency of an ontology is a relevant issue for the work developed in this thesis. This idea is considered in Section 10.2 of Chapter 10.

G-OWL: GRAPHICAL WEB ONTOLOGY LANGUAGE

This appendix presents G-OWL, a graphical language used in this thesis to illustrate the ontology. The aim of this appendix is to give readers the knowledge necessary for understanding the G-OWL figures of the manuscript. Knowledge of OWL is a prerequisite for reading this chapter (see Appendix A).

G-OWL was designed by Michel Héon to represent graphically ontologies implemented using OWL. It is compatible with OWL 2, the last version of OWL. G-OWL is a formal language with a clearly defined grammar. This grammar is made of a few number of symbols which facilitates the reading and the understanding of G-OWL models. This formalism can be used in several stages of the life cycle of an ontology. First, during the conception of it, G-OWL allows the development team to have an overall view of the ontology in a graphical format. Second, during the sharing of the ontology with other people involved in a project or the community, the use of a graphical format facilitates the understanding of the ontology and therefore the appropriation of the components composing it. In this thesis, G-OWL is used in Chapter 7 and Chapter 8 to present the ORD2I ontology and its use.

The use of G-OWL despite of other formalisms was motivated by several reasons. G-OWL is more concise and clearer than the serialisations of OWL ontologies such as RDF/XML, OWL/XML and Turtle. These textual serialisations are not easy to read and they do not allow to quickly have an overview of the ontology. However, G-OWL can not substitute for serialisation in due form of ontology, especially in terms of accuracy. For this reason, Appendix C proposes a serialisation of the ontology using Turtle. Unlike, graphical language for conception such as UML, G-OWL is designed specifically for OWL. Thus, it integrates all the features of this language. Finally, G-OWL was preferred to another graphical ontology language called V-OWL¹. As G-OWL, this language is specifically designed for OWL and thus has the same advantages as G-OWL. However, V-OWL use colours as part of its

¹<http://vowl.visualdataweb.org/>

specification. This feature is not always compatible with the academic publication criteria (black and white printing).

To get the complete specification of G-OWL, the readers can consult the following references:

- Héon, M. (2014). G-owl: Vers une syntaxe graphique de l'owl humainement lisible "human readable", <http://fr.slideshare.net/michelheon/\penalty\z@gowl-graphical-web-ontology-language>
- Héon, M. and Nkambou, R. (2013). G-owl: Vers un langage de modélisation graphique, polymorphe et typé pour la construction d'une ontologie dans la notation owl. In *IC-24èmes Journées francophones d'Ingénierie des Connaissances*

B.1/ G-OWL SYMBOLS

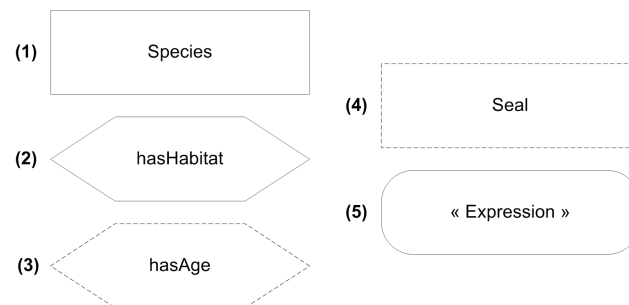


Figure B.1: Symbols of entities in G-OWL

G-OWL uses five different shapes to represent the entities of the ontology. Figure B.1 illustrates the correspondence between the different shapes and the entities of the ontology. The proposed language allows to declare concepts (see (1) in Figure B.1), objects properties (2), datatype properties (3), individuals (4) and expressions/constraints (5). To link these entities, G-OWL introduces several types of properties illustrated in Figure B.2. The single arrow S (see (6) in Figure B.2) allows to describe a hierarchical relationship between two entities (classes, properties, etc.). In the example, the two concepts *Fox* and *Dog* are subclasses of the *Canidae* class. The double arrow S (7) links two entities that are equivalent as the concepts *Dog* and *Doggie* in the example. The link I (8) is used to indicate the instantiation of a class. In the example, *Snoopy* is an individual belonging to *Dog*. The link A (9) specifies the axiomatisation of a property (a declaration of a domain or a range for example). In the example, *hasHabitat* is an object property which have a domain (*Species*) and a range (*Habitat*). Finally, untyped links (10) are used to specify predicates (i.e. to instantiate properties). In the example, two individuals (*Snoopy* and *Roof of the doghouse*) are linked with the predicate *hasHabitat*.

B.2/ AXIOMS OF CLASSES AND INDIVIDUALS

In our ontology, we used only two axioms for classes and individuals. The first one is the subsumption of two classes as represented in part (6) of Figure B.2. The second one is the instantiation of a class by an individual, illustrated in part (8) of Figure B.2.

B.3/ AXIOMS OF PROPERTIES

Regarding properties, G-OWL allows to model hierarchy of object or datatype properties using the arrow S. In the example (11) of Figure B.3, the properties *hasGenus* and *hasCoat* are defined as subproperties of the *hasCharacteristic* property. As shown before in the example (9), a domain and a range can be defined for a given object property using the link A.

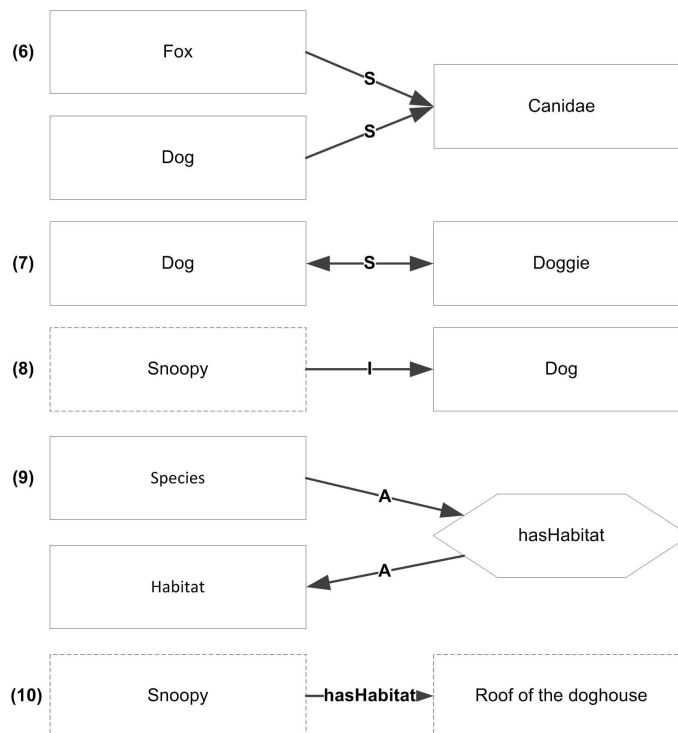


Figure B.2: Symbols of properties in G-OWL

In addition and as shown in the example (10), a property can also be used to construct a predicate linking two individuals. These axioms can also be used with datatype properties as illustrated in the examples (12) and (13) where a domain and a range are defined for the *hasWeight* datatype property and where this property is used to link the individual *Snoopy* with its weight.

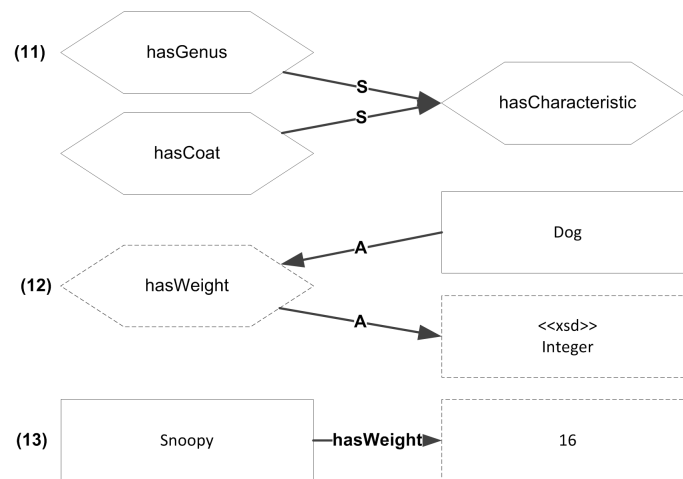


Figure B.3: Axioms of properties in G-OWL

B.4/ CONSTRAINTS ON PROPERTIES

To conclude, we define several constraints on the properties of the ORD2I ontology. G-OWL allows to model all these constraints using the graphical representations illustrated in Figure B.4. First, constraints of cardinality can be defined in properties as shown in the example (14). In this example, an exact cardinality (equals to one) is defined for the *hasName* property regarding the *Dog* class in addition to a max cardinality defined for the *hasParent* property regarding the *Animal* class. Second, various characteristics can be used on properties such as symmetry, transitivity, functionality, etc. In our ontology, we use transitive properties (illustrated in the example (15)) and symmetric properties (shown in example (16)).

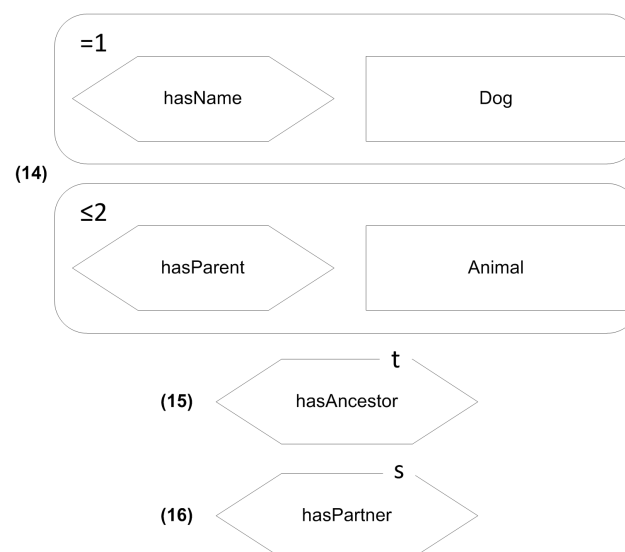


Figure B.4: Constraints on properties in G-OWL

B.5/ CONCLUSION

In this appendix, a language, called G-OWL, for the graphical representation of ontologies was presented. This language has several advantages, including its clarity (because of its graphical nature), its level of formality and its completeness with respect to the OWL features (OWL 2 included). G-OWL is used in this manuscript to illustrate the aspects inherent to the ORD2I ontology and its use. G-OWL can model other features of OWL than those presented in this appendix as the complex class expressions (disjunction, union, intersection, enumeration, etc.), the restrictions on the properties (universal and existential quantification, restriction on values, etc.) or assertions on individuals, etc. However, these features are not used in our work and they are therefore not presented here. We invite the readers to refer to the above-cited reference for more details on these G-OWL features.

SERIALISATION OF THE ORD2I ONTOLOGY

In this appendix, the ORD2I ontology is presented in its entirety. In Listing C.1, a serialisation of it is shown. This serialisation is made using the Turtle language presented in Appendix A. This language is used because it allows to represent ontologies in a clear and concise manner. It is more concise than all the other formats to serialise ontologies, including XML format such as OWL/XML.

```
@prefix : <http://www.w3.org/2002/07/owl#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix xml: <http://www.w3.org/XML/1998/namespace> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@base <http://checksem.u-bourgogne.fr/ord2i> .

<http://checksem.u-bourgogne.fr/ord2i> rdf:type :Ontology ;
    :imports <http://checksem.u-bourgogne.fr/timeRL> ;
    :versionIRI <http://checksem.u-bourgogne.fr/ord2i/1.0.0> .
#####
#
#   Classes
#
#####
### http://checksem.u-bourgogne.fr/ord2i#Account
:Account rdf:type :Class ;
    rdfs:subClassOf :Object .
### http://checksem.u-bourgogne.fr/ord2i#Affiliation
:Affiliation rdf:type :Class .
### http://checksem.u-bourgogne.fr/ord2i#Annotation
:Annotation rdf:type :Class ;
    rdfs:subClassOf :WebObject .
### http://checksem.u-bourgogne.fr/ord2i#ArchiveFile
:ArchiveFile rdf:type :Class ;
    rdfs:subClassOf :File .
### http://checksem.u-bourgogne.fr/ord2i#Bookmark
:Bookmark rdf:type :Class ;
    rdfs:subClassOf :WebObject .
### http://checksem.u-bourgogne.fr/ord2i#Call
:Call rdf:type :Class ;
```

```

    rdfs:subClassOf :Communication .
### http://checksem.u-bourgogne.fr/ord2i#ChatMessage
:ChatMessage rdf:type :Class ;
    rdfs:subClassOf :Communication .
### http://checksem.u-bourgogne.fr/ord2i#ComAccount
:ComAccount rdf:type :Class ;
    rdfs:subClassOf :Account .
### http://checksem.u-bourgogne.fr/ord2i#Communication
:Communication rdf:type :Class ;
    rdfs:subClassOf :Object .
### http://checksem.u-bourgogne.fr/ord2i#Contribution
:Contribution rdf:type :Class .
### http://checksem.u-bourgogne.fr/ord2i#Cookie
:Cookie rdf:type :Class ;
    rdfs:subClassOf :WebObject .
### http://checksem.u-bourgogne.fr/ord2i#Correlation
:Correlation rdf:type :Class .
### http://checksem.u-bourgogne.fr/ord2i#EmailMessage
:EmailMessage rdf:type :Class ;
    rdfs:subClassOf :Communication .
### http://checksem.u-bourgogne.fr/ord2i#Entity
:Entity rdf:type :Class .
### http://checksem.u-bourgogne.fr/ord2i#Event
:Event rdf:type :Class ;
    rdfs:subClassOf :Entity .
### http://checksem.u-bourgogne.fr/ord2i#ExeFile
:ExeFile rdf:type :Class ;
    rdfs:subClassOf :File .
### http://checksem.u-bourgogne.fr/ord2i#File
:File rdf:type :Class ;
    rdfs:subClassOf :Object .
### http://checksem.u-bourgogne.fr/ord2i#FileTransfer
:FileTransfer rdf:type :Class ;
    rdfs:subClassOf :Communication .
### http://checksem.u-bourgogne.fr/ord2i#Form
:Form rdf:type :Class ;
    rdfs:subClassOf :WebObject .
### http://checksem.u-bourgogne.fr/ord2i#FormField
:FormField rdf:type :Class ;
    rdfs:subClassOf :WebObject .
### http://checksem.u-bourgogne.fr/ord2i#ImageFile
:ImageFile rdf:type :Class ;
    rdfs:subClassOf :File .
### http://checksem.u-bourgogne.fr/ord2i#InvestigativeOperation
:InvestigativeOperation rdf:type :Class .
### http://checksem.u-bourgogne.fr/ord2i#Investigator
:Investigator rdf:type :Class ;
    rdfs:subClassOf :Person .
### http://checksem.u-bourgogne.fr/ord2i#Job
:Job rdf:type :Class ;
    rdfs:subClassOf :Object .
### http://checksem.u-bourgogne.fr/ord2i#Link
:Link rdf:type :Class ;
    rdfs:subClassOf :File .
### http://checksem.u-bourgogne.fr/ord2i#LocalVirtualLocation
:LocalVirtualLocation rdf:type :Class ;
    rdfs:subClassOf :VirtualLocation .
### http://checksem.u-bourgogne.fr/ord2i#Location

```



```

:Location rdf:type :Class .
### http://checksem.u-bourgogne.fr/ord2i#Login
:Login rdf:type :Class ;
    rdfs:subClassOf :WebObject .
### http://checksem.u-bourgogne.fr/ord2i#MMS
:MMS rdf:type :Class ;
    rdfs:subClassOf :Communication .
### http://checksem.u-bourgogne.fr/ord2i#OLECF
:OLECF rdf:type :Class ;
    rdfs:subClassOf :File .
### http://checksem.u-bourgogne.fr/ord2i#OSUserAccount
:OSUserAccount rdf:type :Class ;
    rdfs:subClassOf :Account .
### http://checksem.u-bourgogne.fr/ord2i#Object
:Object rdf:type :Class ;
    rdfs:subClassOf :Entity .
### http://checksem.u-bourgogne.fr/ord2i#Organisation
:Organisation rdf:type :Class .
### http://checksem.u-bourgogne.fr/ord2i#PDFFile
:PDFFile rdf:type :Class ;
    rdfs:subClassOf :File .
### http://checksem.u-bourgogne.fr/ord2i#Permission
:Permission rdf:type :Class ;
    rdfs:subClassOf :WebObject .
### http://checksem.u-bourgogne.fr/ord2i#Person
:Person rdf:type :Class ;
    rdfs:subClassOf :Subject .
### http://checksem.u-bourgogne.fr/ord2i#PhysicalLocation
:PhysicalLocation rdf:type :Class ;
    rdfs:subClassOf :Location .
### http://checksem.u-bourgogne.fr/ord2i#Preference
:Preference rdf:type :Class ;
    rdfs:subClassOf :WebObject .
### http://checksem.u-bourgogne.fr/ord2i#Process
:Process rdf:type :Class ;
    rdfs:subClassOf :Subject .
### http://checksem.u-bourgogne.fr/ord2i#RegisterKey
:RegisterKey rdf:type :Class ;
    rdfs:subClassOf :Object .
### http://checksem.u-bourgogne.fr/ord2i#RemoteVirtualLocation
:RemoteVirtualLocation rdf:type :Class ;
    rdfs:subClassOf :VirtualLocation .
### http://checksem.u-bourgogne.fr/ord2i#SMS
:SMS rdf:type :Class ;
    rdfs:subClassOf :Communication .
### http://checksem.u-bourgogne.fr/ord2i#Subject
:Subject rdf:type :Class ;
    rdfs:subClassOf :Entity .
### http://checksem.u-bourgogne.fr/ord2i#Tool
:Tool rdf:type :Class .
### http://checksem.u-bourgogne.fr/ord2i#UnixFile
:UnixFile rdf:type :Class ;
    rdfs:subClassOf :File .
### http://checksem.u-bourgogne.fr/ord2i#UnixUserAccount
:UnixUserAccount rdf:type :Class ;
    rdfs:subClassOf :OSUserAccount .
### http://checksem.u-bourgogne.fr/ord2i#VirtualLocation
:VirtualLocation rdf:type :Class ;

```

```

    rdfs:subClassOf :Location .
### http://checksem.u-bourgogne.fr/ord2i#WebObject
:WebObject rdf:type :Class ;
    rdfs:subClassOf :Object .
### http://checksem.u-bourgogne.fr/ord2i#WebResource
:WebResource rdf:type :Class ;
    rdfs:subClassOf :WebObject .
### http://checksem.u-bourgogne.fr/ord2i#Webpage
:Webpage rdf:type :Class ;
    rdfs:subClassOf :WebResource .
### http://checksem.u-bourgogne.fr/ord2i#Website
:Website rdf:type :Class ;
    rdfs:subClassOf :WebObject .
### http://checksem.u-bourgogne.fr/ord2i#WinExeFile
:WinExeFile rdf:type :Class ;
    rdfs:subClassOf :WinFile .
### http://checksem.u-bourgogne.fr/ord2i#WinFile
:WinFile rdf:type :Class ;
    rdfs:subClassOf :File .
### http://checksem.u-bourgogne.fr/ord2i#WinUserAccount
:WinUserAccount rdf:type :Class ;
    rdfs:subClassOf :OSUserAccount .
#####
#
#   Object Properties
#
#####
### http://checksem.u-bourgogne.fr/ord2i#actedOnBehalfOf
:actedOnBehalfOf rdf:type :ObjectProperty ,
    :TransitiveProperty ;
    rdfs:domain :Investigator ;
    rdfs:range :Investigator .
### http://checksem.u-bourgogne.fr/ord2i#creates
:creates rdf:type :ObjectProperty ;
    rdfs:subPropertyOf :interacts .
### http://checksem.u-bourgogne.fr/ord2i#hasCorrelatedEvent
:hasCorrelatedEvent rdf:type :ObjectProperty ;
    rdfs:domain :Correlation ;
    rdfs:range :Event .
### http://checksem.u-bourgogne.fr/ord2i#hasDateTime
:hasDateTime rdf:type :ObjectProperty ;
    rdfs:domain :Event, :InvestigativeOperation ;
    rdfs:range <http://www.w3.org/2006/time#Interval> .
### http://checksem.u-bourgogne.fr/ord2i#hasInvestigator
:hasInvestigator rdf:type :ObjectProperty ;
    rdfs:domain :Affiliation, :Contribution ;
    rdfs:range :Investigator .
### http://checksem.u-bourgogne.fr/ord2i#hasLocation
:hasLocation rdf:type :ObjectProperty ;
    rdfs:range :Location ;
    rdfs:domain :Object .
### http://checksem.u-bourgogne.fr/ord2i#hasOrganisation
:hasOrganisation rdf:type :ObjectProperty ;
    rdfs:domain :Affiliation ;
    rdfs:range :Organisation .
### http://checksem.u-bourgogne.fr/ord2i#hasPhysicalLocation
:hasPhysicalLocation rdf:type :ObjectProperty ;
    rdfs:domain :Event, :InvestigativeOperation, :Organisation ;

```

```

    rdfs:range :PhysicalLocation ;
    rdfs:subPropertyOf :hasLocation .
### http://checksem.u-bourgogne.fr/ord2i#hasPreviousPageVisited
:hasPreviousPageVisited rdfs:type :ObjectProperty ;
    rdfs:range :Webpage ;
    rdfs:domain :Webpage ;
    rdfs:subPropertyOf :relatedTo .
### http://checksem.u-bourgogne.fr/ord2i#hasReceiver
:hasReceiver rdfs:type :ObjectProperty ;
    rdfs:domain :Call, :ChatMessage, :FileTransfer, :MMS ;
    rdfs:range :Person ;
    rdfs:domain :SMS ;
    rdfs:subPropertyOf :relatedTo .
### http://checksem.u-bourgogne.fr/ord2i#hasSender
:hasSender rdfs:type :ObjectProperty ;
    rdfs:domain :Call, :ChatMessage, :FileTransfer, :MMS ;
    rdfs:range :Person ;
    rdfs:domain :SMS ;
    rdfs:subPropertyOf :relatedTo .
### http://checksem.u-bourgogne.fr/ord2i#interacts
:interacts rdfs:type :ObjectProperty ;
    rdfs:domain :Event ;
    rdfs:range :Object .
### http://checksem.u-bourgogne.fr/ord2i#isComposedOf
:isComposedOf rdfs:type :ObjectProperty ,
    :TransitiveProperty ;
    rdfs:range :Event ;
    rdfs:domain :Event ;
    rdfs:subPropertyOf :links .
### http://checksem.u-bourgogne.fr/ord2i#isIdentifiedBy
:isIdentifiedBy rdfs:type :ObjectProperty ;
    rdfs:domain :Entity ;
    rdfs:range :InvestigativeOperation .
### http://checksem.u-bourgogne.fr/ord2i#isInvolved
:isInvolved rdfs:type :ObjectProperty ;
    rdfs:subPropertyOf :tookPartIn .
### http://checksem.u-bourgogne.fr/ord2i#isMadeFor
:isMadeFor rdfs:type :ObjectProperty ;
    rdfs:domain :Contribution ;
    rdfs:range :InvestigativeOperation .
### http://checksem.u-bourgogne.fr/ord2i#isPerformedWith
:isPerformedWith rdfs:type :ObjectProperty ;
    rdfs:domain :InvestigativeOperation ;
    rdfs:range :Tool .
### http://checksem.u-bourgogne.fr/ord2i#isSupportedBy
:isSupportedBy rdfs:type :ObjectProperty ;
    rdfs:range :Entity ;
    rdfs:domain :InvestigativeOperation .
### http://checksem.u-bourgogne.fr/ord2i#links
:links rdfs:type :ObjectProperty ;
    rdfs:range :Event ;
    rdfs:domain :Event .
### http://checksem.u-bourgogne.fr/ord2i#modifies
:modifies rdfs:type :ObjectProperty ;
    rdfs:subPropertyOf :interacts .
### http://checksem.u-bourgogne.fr/ord2i#relatedTo
:relatedTo rdfs:type :ObjectProperty, :SymmetricProperty, :TransitiveProperty ;
    rdfs:range :Object ;

```

```

    rdfs:domain :Object , :Subject ;
    rdfs:range :Subject .
### http://checksem.u-bourgogne.fr/ord2i#removes
:removes rdfs:type :ObjectProperty ;
    rdfs:subPropertyOf :interacts .
### http://checksem.u-bourgogne.fr/ord2i#tookPartIn
:tookPartIn rdfs:type :ObjectProperty ;
    rdfs:range :Event ;
    rdfs:domain :Subject .
### http://checksem.u-bourgogne.fr/ord2i#undergoes
:undergoes rdfs:type :ObjectProperty ;
    rdfs:subPropertyOf :tookPartIn .
### http://checksem.u-bourgogne.fr/ord2i#uses
:uses rdfs:type :ObjectProperty ;
    rdfs:subPropertyOf :interacts .
#####
#
#   Data properties
#
#####
### http://checksem.u-bourgogne.fr/ord2i#hasAdress
:hasAdress rdfs:type :DatatypeProperty ;
    rdfs:domain :PhysicalLocation ;
    rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasAnnotation
:hasAnnotation rdfs:type :DatatypeProperty ;
    rdfs:domain :Annotation ;
    rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasAuthor
:hasAuthor rdfs:type :DatatypeProperty ;
    rdfs:domain :OLECF ;
    rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasBookmarkType
:hasBookmarkType rdfs:type :DatatypeProperty ;
    rdfs:domain :Bookmark ;
    rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasCity
:hasCity rdfs:type :DatatypeProperty ;
    rdfs:domain :PhysicalLocation ;
    rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasCmdArguments
:hasCmdArguments rdfs:type :DatatypeProperty ;
    rdfs:domain :Link ;
    rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasComments
:hasComments rdfs:type :DatatypeProperty ;
    rdfs:domain :OLECF ;
    rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasCountry
:hasCountry rdfs:type :DatatypeProperty ;
    rdfs:domain :PhysicalLocation ;
    rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasDescription
:hasDescription rdfs:type :DatatypeProperty ;
    rdfs:domain :Entity , :InvestigativeOperation , :Organisation ;
    rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasDriveLetter
:hasDriveLetter rdfs:type :DatatypeProperty ;

```

```

    rdfs:domain :LocalVirtualLocation ;
    rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasEKBasedCorrelation
:hasEKBasedCorrelation rdfs:type :DatatypeProperty ;
    rdfs:domain :Correlation ;
    rdfs:range xsd:double .
### http://checksem.u-bourgogne.fr/ord2i#hasEmail
:hasEmail rdfs:type :DatatypeProperty ;
    rdfs:domain :Person ;
    rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasEventSubtype
:hasEventSubtype rdfs:type :DatatypeProperty ;
    rdfs:domain :Event ;
    rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasEventType
:hasEventType rdfs:type :DatatypeProperty ;
    rdfs:domain :Event ;
    rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasExtension
:hasExtension rdfs:type :DatatypeProperty ;
    rdfs:domain :VirtualLocation ;
    rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasFileAttributes
:hasFileAttributes rdfs:type :DatatypeProperty ;
    rdfs:domain :Link ;
    rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasFilename
:hasFilename rdfs:type :DatatypeProperty ;
    rdfs:domain :VirtualLocation ;
    rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasFirstname
:hasFirstname rdfs:type :DatatypeProperty ;
    rdfs:domain :Person ;
    rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasFolder
:hasFolder rdfs:type :DatatypeProperty ;
    rdfs:domain :Bookmark ;
    rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasFrequency
:hasFrequency rdfs:type :DatatypeProperty ;
    rdfs:domain :Job ;
    rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasHTTPHeader
:hasHTTPHeader rdfs:type :DatatypeProperty ;
    rdfs:domain :WebResource ;
    rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasHostname
:hasHostname rdfs:type :DatatypeProperty ;
    rdfs:domain :RemoteVirtualLocation ;
    rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasID
:hasID rdfs:type :DatatypeProperty ;
    rdfs:domain :Entity , :InvestigativeOperation ;
    rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasIconLocation
:hasIconLocation rdfs:type :DatatypeProperty ;
    rdfs:domain :Link ;
    rdfs:range xsd:string .

```

```

### http://checksem.u-bourgogne.fr/ord2i#hasKeywords
:hasKeywords rdf:type :DatatypeProperty ;
  rdfs:domain :OLECF ;
  rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasLastName
:hasLastName rdf:type :DatatypeProperty ;
  rdfs:domain :Person ;
  rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasMachineName
:hasMachineName rdf:type :DatatypeProperty ;
  rdfs:domain :PhysicalLocation ;
  rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasMessage
:hasMessage rdf:type :DatatypeProperty ;
  rdfs:domain :ChatMessage ;
  rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasName
:hasName rdf:type :DatatypeProperty ;
  rdfs:domain :Cookie, :Organisation, :Process, :Tool ;
  rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasObjectCorrelation
:hasObjectCorrelation rdf:type :DatatypeProperty ;
  rdfs:domain :Correlation ;
  rdfs:range xsd:double .
### http://checksem.u-bourgogne.fr/ord2i#hasOverallCorrelation
:hasOverallCorrelation rdf:type :DatatypeProperty ;
  rdfs:domain :Correlation ;
  rdfs:range xsd:double .
### http://checksem.u-bourgogne.fr/ord2i#hasPageCounter
:hasPageCounter rdf:type :DatatypeProperty ;
  rdfs:domain :OLECF ;
  rdfs:range xsd:integer .
### http://checksem.u-bourgogne.fr/ord2i#hasParameters
:hasParameters rdf:type :DatatypeProperty ;
  rdfs:domain :Job ;
  rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasPath
:hasPath rdf:type :DatatypeProperty ;
  rdfs:domain :LocalVirtualLocation ;
  rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasPhone
:hasPhone rdf:type :DatatypeProperty ;
  rdfs:domain :Person ;
  rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasPosition
:hasPosition rdf:type :DatatypeProperty ;
  rdfs:domain :Affiliation ;
  rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasPseudo
:hasPseudo rdf:type :DatatypeProperty ;
  rdfs:domain :Account, :Person ;
  rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasRelativePath
:hasRelativePath rdf:type :DatatypeProperty ;
  rdfs:domain :Link ;
  rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasRevisionNumber
:hasRevisionNumber rdf:type :DatatypeProperty ;

```

```

    rdfs:domain :OLECF ;
    rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasRole
:hasRole rdf:type :DatatypeProperty ;
    rdfs:domain :Contribution ;
    rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasSecurity
:hasSecurity rdf:type :DatatypeProperty ;
    rdfs:domain :OLECF ;
    rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasSerial
:hasSerial rdf:type :DatatypeProperty ;
    rdfs:domain :Link ;
    rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasShortDescription
:hasShortDescription rdf:type :DatatypeProperty ;
    rdfs:domain :Entity , :InvestigativeOperation , :Organisation ;
    rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasSize
:hasSize rdf:type :DatatypeProperty ;
    rdfs:domain :File , :WebResource ;
    rdfs:range xsd:integer .
### http://checksem.u-bourgogne.fr/ord2i#hasSourceName
:hasSourceName rdf:type :DatatypeProperty ;
    rdfs:domain :InvestigativeOperation ;
    rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasSourceType
:hasSourceType rdf:type :DatatypeProperty ;
    rdfs:domain :InvestigativeOperation ;
    rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasStatus
:hasStatus rdf:type :DatatypeProperty ;
    rdfs:domain :Event ;
    rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasSubjectCorrelation
:hasSubjectCorrelation rdf:type :DatatypeProperty ;
    rdfs:domain :Correlation ;
    rdfs:range xsd:double .
### http://checksem.u-bourgogne.fr/ord2i#hasTechniqueName
:hasTechniqueName rdf:type :DatatypeProperty ;
    rdfs:domain :InvestigativeOperation ;
    rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasTechniqueType
:hasTechniqueType rdf:type :DatatypeProperty ;
    rdfs:domain :InvestigativeOperation ;
    rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasTemplate
:hasTemplate rdf:type :DatatypeProperty ;
    rdfs:domain :OLECF ;
    rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasTemporalCorrelation
:hasTemporalCorrelation rdf:type :DatatypeProperty ;
    rdfs:domain :Correlation ;
    rdfs:range xsd:double .
### http://checksem.u-bourgogne.fr/ord2i#hasTitle
:hasTitle rdf:type :DatatypeProperty ;
    rdfs:domain :Bookmark , :ChatMessage , :OLECF , :Webpage ;
    rdfs:range xsd:string .

```

```

### http://checksem.u-bourgogne.fr/ord2i#hasToolType
:hasToolType rdf:type :DatatypeProperty ;
  rdfs:domain :Tool ;
  rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasTopic
:hasTopic rdf:type :DatatypeProperty ;
  rdfs:domain :OLECF ;
  rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasTruthfulness
:hasTruthfulness rdf:type :DatatypeProperty ;
  rdfs:domain :InvestigativeOperation ;
  rdfs:range xsd:double .
### http://checksem.u-bourgogne.fr/ord2i#hasURL
:hasURL rdf:type :DatatypeProperty ;
  rdfs:domain :RemoteVirtualLocation, :WebResource ;
  rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasURLWebpage
:hasURLWebpage rdf:type :DatatypeProperty ;
  rdfs:domain :Bookmark ;
  rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasUsageCounter
:hasUsageCounter rdf:type :DatatypeProperty ;
  rdfs:domain :WebResource, :WinExeFile ;
  rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasVersion
:hasVersion rdf:type :DatatypeProperty ;
  rdfs:domain :Process, :Tool ;
  rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#hasWordCounter
:hasWordCounter rdf:type :DatatypeProperty ;
  rdfs:domain :OLECF ;
  rdfs:range xsd:integer .
### http://checksem.u-bourgogne.fr/ord2i#hasZipcode
:hasZipcode rdf:type :DatatypeProperty ;
  rdfs:domain :PhysicalLocation ;
  rdfs:range xsd:string .
### http://checksem.u-bourgogne.fr/ord2i#isHTTPOnly
:isHTTPOnly rdf:type :DatatypeProperty ;
  rdfs:domain :Cookie ;
  rdfs:range xsd:boolean .
### http://checksem.u-bourgogne.fr/ord2i#isPersistent
:isPersistent rdf:type :DatatypeProperty ;
  rdfs:domain :Cookie ;
  rdfs:range xsd:boolean .
### http://checksem.u-bourgogne.fr/ord2i#topDataProperty
:topDataProperty rdf:type :DatatypeProperty .

```

Listing C.1: Turtle serialisation of the ORD2I Ontology

INFORMATION SOURCES

In Chapter 8, the ANNALIST architecture is presented. This architecture is made of several layers, each carrying out some of the tasks composing the reconstruction of events. Among these tasks, the extraction of the knowledge contained in each source of information that can be found in a digital crime scene is crucial. This step aims to extract information about past events in an efficient and exhaustive way. The main goal of this appendix is to present all sources of information used as input of ANNALIST. The second goal is to explain how this information is processed by the extraction layer. To ensure the conciseness of this thesis, the process instantiating the ontology is not presented in this appendix. However, the reader can refer to Section 8.3 for an explanation of the instantiation of ORD2I.

This appendix is structured in the following way. Section D.1 lists the information sources used as input of ANNALIST. Each of these sources is then the subject of a detailed presentation in the following sections. For each source, its characteristics and its relevance regarding the objectives of the investigation are first presented. The information about each type of events regarding this source (generated by Plaso) are then given. As presented in Section 8.2, extraction patterns are used to extract information inherent to each type of events. The patterns related to each information sources are presented in this appendix.

D.1/ INFORMATION SOURCES

As said in Chapter 2, a digital crime scene contains a large number of heterogeneous sources of information. The ANNALIST architecture introduced in Chapter 8 is able to handle a large panel of sources. To achieve this objective, the extraction layer takes advantage of the Plaso toolbox (and more particularly, *log2timeline*). The results produced by Plaso are then filtered and adapted to meet the needs of the upper layers of the architecture.

This section lists the sources of information used as input of ANNALIST. It should be noted that the number of sources managed by ANNALIST is less than the number of sources handle by Plaso. Indeed, the number of information sources has been restricted to reduce the time required to develop the prototype of ANNALIST. Table D.1 gives a comprehensive list of the information sources that can be processed by Plaso (see column "Information source"). For comparison, the sources of information supported by ANNALIST are indicated (✓ if the source is supported and ✗ if not).

Information source	ANNALIST	Information source	ANNALIST
Android	✗	Opera	✗
Application Compatibility Cache	✗	PCAP	✗
ASL securityd logs	✗	PL/SQL	✗
Bencode	✗	Plist	✗
BSM	✗	Popularity contest	✗
Chrome	✓	Recycle Bin	✓
CUPS IPP	✗	Safari	✓
Filestat	✓	selinux	✗
Firefox	●	SkyDrive	✗
Google Analytics	✗	Skype	✓
Google Drive	✓	Symantec	✗
Internet Explorer	●	syslog	✗
Ipod	✗	UTMP	✗
Java IDX	✓	UTMPX	✗
Keychain database	✗	Windows IIS	✗
Link	✓	Win Prefetch	✓
Mac OS	✗	Windows Event Log	✗
MacKeeper	✗	Windows Register	✗
McAfee	✗	Windows Firewall	✗
OLE Compound File	✓	xchatlog	✗
OpenXML	✗	xchatscrollback	✗

Table D.1: Information sources supported by Plaso and ANNALIST

D.2/ FIREFOX

Mozilla Firefox is an open-source web browser maintained by the Mozilla Foundation. This browser stores a lot of information about users in SQLite databases, log files and registry. Firefox allows the creation of several user profiles (in addition to the default profile). Each profile has its own bookmarks and its own history store in a dedicated folder. The file profiles.ini contains information about the user profiles existing on the machine (including the location of the SQLite and log files related to each profile). Thus, this file can be used as a starting point for the study of the information generated by Firefox. The information stored in the SQLite databases describes the activities of the user. This includes the pages visited by the user, the cookies, the downloads, the filled forms, etc. This source of information is therefore very relevant to get a better understanding of the activities of a given user on the Web.

The formatting of data generated by Firefox is made by the script firefox.py of Plaso. The three types of entries that can be generated using this script are shown in Listing D.1.

```

date, time, timezone, MACB, source, sourcetype, type, user, host, short, desc, version,
filename, inode, notes, format, extra
11/10/2014, 08:28:57, UTC, ...B, WEBHIST, Firefox History, Creation Time, -,
WIN-I51P7DIKOO0, Bookmark Annotation: Etiquettes recentes, Bookmark Annotation:
[RecentTags] to bookmark [Etiquettes recentes]
(place:type=6&sort=14&maxResults=10), 2,
TSK:/Users/UserX/AppData/Roaming/Mozilla/Firefox/Profiles/z7vp.default/
places.sqlite, 43424, -, sqlite, plugin: firefox_history
11/10/2014, 08:32:44, UTC, ...B, WEBHIST, Firefox History, Creation Time, -,
WIN-I51P7DIKOO0, Bookmarked Yoan Chabot (http://www.yoan-chabot.fr/en/index.php),
Bookmark URL Yoan Chabot (http://www.yoan-chabot.fr/en/index.php) [Yoan Chabot]
visit count 2, 2,
TSK:/Users/UserX/AppData/Roaming/Mozilla/Firefox/Profiles/z7vp.default/
places.sqlite, 43424, -, sqlite, host: N/A plugin: firefox_history
11/10/2014, 08:28:32, UTC, .A., WEBHIST, Firefox History, Page Visited, -,
WIN-I51P7DIKOO0, URL: https://www.mozilla.org/fr/firefox/new,
https://www.mozilla.org/fr/firefox/new (Telecharger Firefox - Navigateur web
gratuit - Mozilla) [count: 1] Host: www.mozilla.org (URL not typed directly)
Transition: LINK, 2,
TSK:/Users/UserX/AppData/Roaming/Mozilla/Firefox/Profiles/z7vp.default/
places.sqlite, 43424, -, sqlite, visit_type: 1 extra: [u'(URL not typed directly)'
u'Transition: LINK'] plugin: firefox_history

```

Listing D.1: Information extracted from Firefox using Plaso

Each entry contains the following information:

- *date*, *time* and *timezone*: time information.
- *MACB* and *type*: information about the type of action. These two fields have the values "...B" and "Creation Time" if the action is the creation of a bookmark or the annotation of a bookmark. They have the values ".A.." and "Page Visited" in the case of the visit of a webpage.
- *source* and *sourcetype*: type of event. The values of these fields are respectively set at "WEBHIST" and "Firefox History".
- *host*: information about the host (i.e. the machine on which the event happens) of the event.

The *desc* field for entries corresponding to the creation of an annotation is made of the following information: the annotation, the name of the bookmark and the URL of the webpage. To extract this information, the following extraction pattern is used:

Extraction pattern ¹. *Bookmark Annotation: [{annotation}] to bookmark [{title}] ({url})*

Regarding entries representing the creation of a bookmark, the *desc* field contains the type of the bookmark, its name, the URL of the related webpage, the title of the webpage and the usage counter of the bookmark. The extraction pattern above is used to extract this information:

Extraction pattern ². *Bookmark {type} {title} ({url}) [{pageTitle}] visit count {visit Counter}*

For entries corresponding to the visit of a webpage, the URL of the visited webpage, its title, its usage counter, its hostname and additional information (webpage previously visited, type of transition, etc.) can be extracted using the following pattern:

Extraction pattern ³. *{url} ({title}) [count: {visitCounter}]] Host: {host} {extra Information}*

D.3/ CHROME

Google Chrome is a proprietary software developed by Google. Like Firefox, Chrome stores the information about the users in a SQLite database. The formatting of data regarding the history of the user is made by the script *chrome.py* of Plaso. The entries that can be generated using this script are shown in Listing D.2.

```
date, time, timezone, MACB, source, sourcetype, type, user, host, short, desc, version,
filename, inode, notes, format, extra
11/10/2014, 08:30:49, UTC, .A., WEBHIST, Chrome History, Page Visited, -,
WIN-I51P7DIKOO0, http://checksem.u-bourgogne.fr/, http://checksem.u-bourgogne.fr/
[count: 0] Host: checksem.u-bourgogne.fr (URL not typed directly - no typed count),
2, TSK:/Users/UserX/AppData/Local/Google/Chrome/User Data/Default/History, 43770,
-, sqlite, plugin: chrome_history
```

```

11/10/2014, 08:36:18, UTC, .A.., WEBHIST, Chrome History, Page Visited, -,
WIN-I51P7DIKOO0, http://www.yoan-chabot.fr/en/index.php (Yoan Chabot),
http://www.yoan-chabot.fr/en/index.php (Yoan Chabot) [count: 0] Host:
www.yoan-chabot.fr Visit from: http://yoan-chabot.fr/ (Yoan Chabot) (URL not typed
directly - no typed count), 2, TSK:/Users/UserX/AppData/Roaming/Opera
Software/Opera Stable/History, 45905, -, sqlite, plugin: chrome_history
11/10/2014, 08:31:18, UTC,...B, WEBHIST, Chrome History, File Downloaded, -,
WIN-I51P7DIKOO0, C:\Users\UserX\Pictures\arton245-ce9a0.png downloaded (42929
bytes), http://checksem.u-bourgogne.fr/www/local/cache-vignettes/arton245-ce9a0.png
(C:\Users\UserX\Pictures\arton245-ce9a0.png). Received: 42929 bytes out of: 42929
bytes., 2, TSK:/Users/UserX/AppData/Local/Google/Chrome/User Data/Default/History,
43770, -, sqlite, plugin: chrome_history

```

Listing D.2: Information extracted from Google Chrome History using Plaso

Each entry contains the following information:

- *date*, *time* and *timezone*: information indicating the date of the visit or the date of the download.
- *MACB* and *type*: information about the type of action. These two fields have the values "...B" and "File Downloaded" if the action is the download of a file. They have the values ".A.." and "Page Visited" in the case of the visit of a webpage.
- *source* and *sourcetype*: type of event. The values of these fields are respectively set at "WEBHIST" and "Chrome History".
- *host*: information about the host (i.e. the machine on which the event happens) of the event.

For entries corresponding to the visit of a webpage, the URL of the visited webpage, its title, its usage counter, its hostname and additional information (webpage previously visited, the type of transition (for example, URL typed directly into the search bar), etc.) can be extracted from the *desc* field using the following pattern:

Extraction pattern 4. `{url} ({title}) [count: {visitCounter}] Host:{host} Visit from: {fromVisit} Visit Source: [{visitSource}] {extraInformation}`

The *desc* field for entries corresponding to a download is made of the following information: the URL of the remote resource, the local path used to store the resource, the volume of data already downloaded and the total size of the resource. To extract this information, the following extraction pattern is used:

Extraction pattern 5. `{url} ({localPath}) Received: {receivedBytes} bytes out of: {size} bytes.`

The formatting of data regarding the cookies is made by the script `chrome_cookies.py` of Plaso. The entries that can be generated using this script are shown in Listing D.3.

```

date, time, timezone, MACB, source, sourcetype, type, user, host, short, desc, version,
filename, inode, notes, format, extra

```

```

11/10/2014, 08:35:50, UTC, ...B, WEBHIST, Chrome Cookies, Creation Time, -,
WIN-I51P7DIKOO0, opera.com (.ga), http://opera.com/ (.ga) Flags: [HTTP only] =
False [Persistent] = True, 2, TSK:/Users/UserX/AppData/Roaming/Opera Software/Opera
Stable/Cookies, 45903, -, sqlite, secure: False host: opera.com path: / plugin:
chrome_cookies
11/10/2014, 08:35:50, UTC, .A., WEBHIST, Chrome Cookies, Last Access Time, -,
WIN-I51P7DIKOO0, opera.com (.gat), http://opera.com/ (.gat) Flags: [HTTP only] =
False [Persistent] = True, 2, TSK:/Users/UserX/AppData/Roaming/Opera Software/Opera
Stable/Cookies, 45903, -, sqlite, secure: False host: opera.com path: / plugin:
chrome_cookies
11/10/2014, 08:40:01, UTC, ...., WEBHIST, Chrome Cookies, Cookie Expires, -,
WIN-I51P7DIKOO0, sciencedirect.com (optimizelyPendingLogEvents),
http://sciencedirect.com/ (optimizelyPendingLogEvents) Flags: [HTTP only] = False
[Persistent] = True, 2, TSK:/Users/UserX/AppData/Roaming/Opera Software/Opera
Stable/Cookies, 45903, -, sqlite, secure: False host: sciencedirect.com path: /
plugin: chrome_cookies

```

Listing D.3: Information extracted from Google Chrome Cookies using Plaso

Each entry contains the following information:

- *date*, *time* and *timezone*: information indicating the date of creation, consultation or expiration of the cookie.
- *MACB* and *type*: information about the type of action. These two fields have the values "...B" and "Creation Time" if the action is the creation of a cookie. They have the values ".A." and "Last Access Time" in the case of the use of a cookie and the values "...." and "Cookies Expires" when a cookie expires.
- *source* and *sourcetype*: type of event. The values of these fields are respectively set at "WEBHIST" and "Chrome Cookies".
- *host*: information about the host (i.e. the machine on which the event happens) of the event.

The *desc* field of entries, regardless of their type, contain the following information: the URL of the page related to the cookie, the name of the cookie and two flags. For the extraction of this information, the extraction pattern above is used:

Extraction pattern ⁶. `{url} ({cookieName}) Flags: [HTTP only] = {httpOnly} [Persistent] = {persistent}`

The formatting of data regarding the cached files is made by the script `chrome_cache.py` of Plaso. The entries that can be generated using this script are shown in Listing D.4.

```

date, time, timezone, MACB, source, sourcetype, type, user, host, short, desc, version,
filename, inode, notes, format, extra
11/10/2014, 08:30:33, UTC, ...B, WEBHIST, Chrome Cache, Creation Time, -,
WIN-I51P7DIKOO0, Original URL: https://ssl.gstatic.com/accounts/ui/logo_2x.png,
Original URL: https://ssl.gstatic.com/accounts/ui/logo_2x.png, 2,
TSK:/Users/UserX/AppData/Local/Google/Chrome/User
Data/Default/Storage/ext/chrome-signin/def/Cache/index, 44530, -, chrome_cache,

```

Listing D.4: Information extracted from Google Chrome Cache using Plaso

Each entry contains the following information:

- *date*, *time* and *timezone*: information indicating the date at which the file was cached.
- *MACB* and *type*: information about the type of action. These two fields have the values "...B" and "Creation Time".
- *source* and *sourcetype*: type of event. The values of these fields are respectively set at "WEBHIST" and "Chrome Cache".
- *host*: information about the host (i.e. the machine on which the event happens) of the event.

The *desc* field contains only the URL of the cached resource. For the extraction of this information, the extraction pattern above is used:

Extraction pattern ⁷. *Original URL*: {url}

D.4/ INTERNET EXPLORER

Microsoft Internet Explorer is a proprietary web browser embedded in every Windows operating system. The formatting of data regarding the cached files is made by the script msiecf.py of Plaso. The entries that can be generated using this script are shown in Listing D.5.

```
date, time, timezone, MACB, source, sourcetype, type, user, host, short, desc, version,
filename, inode, notes, format, extra
11/24/2014, 08:29:18, UTC, ..., WEBHIST, MSIE Cache File URL record, Expiration Time,
-, WIN-I51P7DIKOO0, Location:
http://ts1.mm.bing.net/th?id=VN.68856&pid=4.1&w=120&h=..., Location:
http://ts1.mm.bing.net/th?id=VN.60887pid=4.1&w=120&h=68&rs=1&vt=4&c=7 Number of
hits: 2 Cached file size: 3149 HTTP headers: HTTP/1.1 200 OK - Content-Type:
image/jpeg - Content-Length: 3149 - - ~U:yoan - , 2,
TSK:/Users/UserX/AppData/Local/Microsoft/Windows/Temporary Internet
Files/Low/Content.IE5/index.dat, 43008, -, msiecf, cache_directory_index: 1
recovered: False
12/02/2014, 07:34:15, UTC, M..., WEBHIST, MSIE Cache File URL record, Content
Modification Time, -, WIN-I51P7DIKOO0, Location:
http://img.s-msn.com/tenant/amp/entityid/BBgdOG0.img..., Location:
http://img.s-msn.com/tenant/amp/entityid/BBgdOG0.img Number of hits: 2 Cached file
size: 2535 HTTP headers: HTTP/1.1 200 OK - Content-Type: image/jpeg -
Content-Location: http://img.s-msn.com/tenant/amp/entityid/BBgdOG0 -
X-Source-Length: 343082 - X-Datcenter: northeu - X-ActivityId:
746dbe8e-4b34-489c-a362-c55cc48201dd - X-Instance: Resizer.Web_IN_11 -
X-Deployment: d79a05dfd16748aab85e3663a3b5039e - Timing-Allow-Origin: * -
X-AspNet-Version: 4.0.30319 - X-Powered-By: ASP.NET - Content-Length: 2535 - -
~U:yoan - , 2, TSK:/Users/UserX/AppData/Local/Microsoft/Windows/Temporary Internet
Files/Low/Content.IE5/index.dat, 43008, -, msiecf, cache_directory_index: 0
recovered: False
12/02/2014, 13:16:04, UTC, ..., WEBHIST, MSIE Cache File URL record, Last Checked Time,
-, WIN-I51P7DIKOO0, Location:
https://api.skype.com/users/pr.nicolle/profile/avatar, Location:
https://api.skype.com/users/pr.nicolle/profile/avatar Number of hits: 4 Cached file
```

```

size: 3213 HTTP headers: HTTP/1.1 200 OK - Content-Type: image/jpeg -
Content-Length: 3213 - X-avatar: optimized - X-Avatar-Type: custom -
X-Stratus-Processing-Time: 0.0588 - X-Content-Type-Options: nosniff -
X-Processing-Time: 0.060 - - ~U:yoan -
,2,TSK:/Users/UserX/AppData/Local/Microsoft/Windows/Temporary Internet
Files/Content.IE5/index.dat, 16388, -, msiecf, cache_directory_index: 0 recovered:
False
12/02/2014, 13:16:05, UTC, .A., WEBHIST, MSIE Cache File URL record, Last Access Time,
-, WIN-I51P7DIKOO0, Location: https://apps.skype.com/countrycode..., Location:
https://apps.skype.com/countrycode Number of hits: 1 Cached file size: 51 HTTP
headers: HTTP/1.1 200 OK - Content-Length: 51 - Content-Type: application/json - -
~U:yoan - , 2, TSK:/Users/UserX/AppData/Local/Microsoft/Windows/Temporary Internet
Files/Content.IE5/index.dat, 16388, -, msiecf, cache_directory_index: 3 recovered:
False

```

Listing D.5: Information extracted from Internet Explorer using Plaso

Each entry contains the following information:

- *date*, *time* and *timezone*: temporal information.
- *MACB* and *type*: information about the type of action. These two fields have the values "... " and "Expiration Time" if the action is the expiration of a cached resource. They have the values "M..." and "Content Modification Time" if the action is the modification of a cached resource. They have the values "... " and "Last Checked Time" if the action is the consultation of a cached resource. They have the values ".A.." and "Last Access Time" if the action is the use of a cached resource.
- *source* and *sourcetype*: type of event. The values of these fields are respectively set at "WEBHIST" and "MSIE Cache File URL record".
- *host*: information about the host (i.e. the machine on which the event happens) of the event.

The *desc* field is made of the following information: the URL of the cached resource, its usage counter, its size and extra information including HTTP headers. For the extraction of this information, the extraction pattern above is used:

Extraction pattern ⁸. *Location: {url} Number of hits: {useCounter} Cached file size: {size} HTTP headers: {httpHeaders} {extraString}*

D.5/ SAFARI

Safari is a proprietary web browser developed by Apple. The formatting of data regarding the history of the user is made by the script `safari.py` of Plaso. The entries that can be generated using this script are shown in Listing D.6.


```

date, time, timezone, MACB, source, sourcetype, type, user, host, short, desc, version,
filename, inode, notes, format, extra
11/10/2014, 08:37:51, UTC, .A.., WEBHIST, Safari History, Last Visited Time, -,
WIN-I51P7DIKOO0, Visited: http://www.apple.com/safari/welcome/ (Apple ) Visit
Count: 1, Visited: http://www.apple.com/safari/welcome/ (Apple ) Visit Count: 1, 2,
TSK:/Users/UserX/AppData/Apple Computer/Safari/History.plist, 47567, -,
plist, plugin: safari_history

```

Listing D.6: Information extracted from Safari using Plaso

Each entry contains the following information:

- *date*, *time* and *timezone*: information indicating the date of the webpage visit.
- *MACB* and *type*: information about the type of action. These two fields have the values "A.." and "Last Visited Time".
- *source* and *sourcetype*: type of event. The values of these fields are respectively set at "WEBHIST" and "Safari History".
- *host*: information about the host (i.e. the machine on which the event happens) of the event.

The *desc* field is made of the following information: the URL of the webpage, its title and its usage counter. For the extraction of this information, the extraction pattern above is used:

Extraction pattern ⁹. *Visited: {url} ({title} - {displayTitle}) Visit Count: {visitCounter}*

D.6/ WIN PREFETCH

The Windows Prefetch folder is used by Windows (since Windows XP) to store data about the last executed programs to help accelerate their future executions. The files located in the Prefetch folder contain the name of the executable, the list of the DLL used by it, a usage counter and a timestamp indicating the date of last execution. The formatting of data regarding the Prefetch folder is done by the script winprefetch.py of Plaso. The entries that can be generated using this script are shown in Listing D.7.

```

date, time, timezone, MACB, source, sourcetype, type, user, host, short, desc, version,
filename, inode, notes, format, extra
11/10/2014, 07:16:26, UTC, .A.., LOG, WinPrefetch, Last Time Executed, -,
WIN-I51P7DIKOO0, WINMAIL.EXE was run 1 time(s), Prefetch [WINMAIL.EXE] was executed
- run count 1 path: \PROGRAM FILES\WINDOWS MAIL\WINMAIL.EXE hash: 0xD6E90604
volume: 1 [serial number: 0x724766A7 device path: \DEVICE\HARDDISKVOLUME1], 2,
TSK:/Windows/Prefetch/WINMAIL.EXE-D6E90604.pf, 15925, -, prefetch,
number_of_volumes: 1 volume_device_paths: [u'\\DEVICE\\HARDDISKVOLUME1']
volume_serial_numbers: [1917281959L] version: 23 prefetch_hash: 3605595652
11/10/2014, 08:29:54, UTC, .A.., LOG, WinPrefetch, Last Time Executed, -,
WIN-I51P7DIKOO0, CHROMESETUP.EXE was run 1 time(s), Prefetch [CHROMESETUP.EXE] was
executed - run count 1 path: \USERS\YOAN\DOWNLOADS\CHROMESETUP.EXE hash: 0xE9793A8D
volume: 1 [serial number: 0x724766A7 device path: \DEVICE\HARDDISKVOLUME1], 2,

```

```

TSK:/Windows/Prefetch/CHROMESETUP.EXE-E9793A8D.pf, 43433, -,
prefetch,number_of_volumes: 1 volume_device_paths: [u'\\DEVICE\\HARDDISKVOLUME1']
volume_serial_numbers: [1917281959L] version: 23 prefetch_hash: 3917036173
11/10/2014, 08:29:54, UTC, .A.., LOG, WinPrefetch, Last Time Executed, -,
WIN-I51P7DIKOO0,GOOGLEUPDATE.EXE was run 1 time(s),Prefetch [GOOGLEUPDATE.EXE] was
executed - run count 1 path:
\USERS\YOAN\APPDATA\LOCAL\TEMP\GUMF804.TMP\GOOGLEUPDATE.EXE hash: 0x6E7D88ED
volume: 1 [serial number: 0x724766A7 device path: \DEVICE\HARDDISKVOLUME1], 2,
TSK:/Windows/Prefetch/GOOGLEUPDATE.EXE-6E7D88ED.pf, 43993, -, prefetch,
number_of_volumes: 1 volume_device_paths: [u'\\DEVICE\\HARDDISKVOLUME1']
volume_serial_numbers: [1917281959L] version: 23 prefetch_hash: 1853720813

```

Listing D.7: Information related to Win Prefetch extracted using Plaso

Each entry contains the following information:

- *date*, *time* and *timezone*: this information indicates the date at which the file was executed and at which the information was added into the Prefetch folder.
- *MACB* and *type*: information about the type of action. These two fields have the values "A.." and "Last Time Executed" meaning that a file was accessed and executed.
- *source* and *sourcetype*: type of event. The values of these fields are respectively set at "LOG" and "WinPrefetch".
- *host*: information about the host (i.e. the machine on which the event happens) of the event.

The *desc* field is made of the following information: the name of the executable, its usage counter, its path, its hash and information about the volume containing it. For the extraction of this information, the extraction pattern above is used:

Extraction pattern ¹⁰. *Prefetch [{executableName}] was executed - run count {run Count} path: {path} hash: 0x{prefetchHash} {infosVolumes}*

D.7/ FILESTAT

The file system of the operating systems keeps, for each file, several timestamps indicating:

- The date of the most recent access (*atime*). This timestamp indicates the date of the most recent reading of the file by a process. If the process keeps the file open for a certain duration, the *atime* does not match with the actual date of last reading.
- The date of the last modification (*mtime*). This timestamp indicates the date of last entry in the file by a process.
- The date of creation of a file or modification of the metadata (*ctime* and *btime*). The meaning of these timestamps differs depending on the operating system. In a Unix system, *ctime* indicates the date of the last modification of the metadata of the file

(e.g. permissions). In a Window system, ctime and btime are used to store the date of creation of a file.

The formatting of data regarding these timestamps is done by the script filestat.py of Plaso. The entries that can be generated using this script are shown in Listing D.8.

```
date, time, timezone, MACB, source, sourcetype, type, user, host, short, desc, version,
filename, inode, notes, format, extra
11/24/2014, 11:47:17, UTC, ..C., FILE, NTFS_DETECT ctime,ctime, -, WIN-I51P7DIKOO0,
/Windows/Prefetch/JXPIINSTALL.EXE-A3782E24.pf,
TSK:/Windows/Prefetch/JXPIINSTALL.EXE-A3782E24.pf, 2,
TSK:/Windows/Prefetch/JXPIINSTALL.EXE-A3782E24.pf, 12831, -, filestat, allocated:
True fs_type: NTFS_DETECT size: [131008L]
11/24/2014, 11:47:22, UTC, ...B, FILE, NTFS_DETECT crtime;atime,crtime;atime, -,
WIN-I51P7DIKOO0,/Windows/Registration/{02D4B3F1}.{919103B8-29AE-4...,
TSK:/Windows/Registration/{02D4B3F1}.{919103B8}.crmlog, 2,
TSK:/Windows/Registration/{02D4B3F1}.{919103B8}.crmlog, 10672, -, filestat,
allocated: True fs_type: NTFS_DETECT size: [1048576L]
11/24/2014, 11:47:24, UTC, M..., FILE, NTFS_DETECT mtime;ctime,mtime;ctime, -,
WIN-I51P7DIKOO0, /Users/UserX/AppData/Local/Temp/_MEI21122/wx._controls_.pyd,
TSK:/Users/UserX/AppData/Local/Temp/_MEI21122/wx._controls_.pyd, 2,
TSK:/Users/UserX/AppData/Local/Temp/_MEI21122/wx._controls_.pyd, 50480, -,
filestat, allocated: True fs_type: NTFS_DETECT size: [1062400L]
11/24/2014, 11:47:59, UTC, .A..., FILE, NTFS_DETECT atime,atime, -, WIN-I51P7DIKOO0,
/Users/UserX/AppData/Local/Microsoft/Windows/Temporary Internet Files/Content...,
TSK:/Users/UserX/AppData/Local/Microsoft/Windows/Temporary Internet
Files/Content.IE5/FEL4CWGD/avatar[3].jpg, 2,
TSK:/Users/UserX/AppData/Local/Microsoft/Windows/Temporary Internet
Files/Content.IE5/FEL4CWGD/avatar[3].jpg, 50021, -, filestat, allocated: True
fs_type: NTFS_DETECT size: [4169L]
```

Listing D.8: Information extracted from the file system using Plaso

Each entry contains the following information:

- *date*, *time* and *timezone*: this information indicates the date of the action (reading, writing or creation).
- *MACB* and *type*: information about the type of action. These two fields are set to "M..." and "mtime;ctime" if a file was modified, ".A.." and "atime" if a file has been consulted, ".C.." and "ctime" if a file was created or its metadata have been modified and "B..." and "crtime;atime" if a file was created.
- *source* and *sourcetype*: type of event. The value of the *source* field is set to "FILE". The value of the *sourcetype* field depends on the file system and the type of operation.
- *host*: information about the host (i.e. the machine on which the event happens) of the event.

The *desc* field contains the path of the file concerned with the operations. For the extraction of this information, the extraction pattern above is used:

Extraction pattern ¹¹. (TSK:[{filePath}].)*

D.8/ GOOGLE DRIVE

Google Drive is a service made available by Google and allowing to store files in the Cloud. Potentially useful information is stored in the machine communicating with this service. This information may help to identify what a given user sent in the cloud and what he has downloaded and saved on the local machine. The formatting of data related to this source is done by the script `gdrive.py` of Plaso. The entries that can be generated using this script are shown in Listing D.9.

```
date, time, timezone, MACB, source, sourcetype, type, user, host, short, desc, version,
filename, inode, notes, format, extra
11/10/2014, 08:45:35, UTC, M..., LOG, Google Drive (local entry), Content Modification
Time, -, WIN-I51P7DIKOO0, %local_sync_root%/Images/photoidentite.bmp, File Path:
%local_sync_root%/Images/photoidentite.bmp Size: 848294, 2,
TSK:/Users/UserX/AppData/Local/Google/Drive/snapshot.db,48581, -, sqlite, plugin:
google_drive
11/07/2014, 07:39:01, UTC, M..., LOG, Google Drive (local entry), Content Modification
Time, -, WIN-I51P7DIKOO0, %local_sync_root%/Rapports/Reunion du 21 novembre -
Copy.txt, File Path: %local_sync_root%/Rapports/Reunion du 21 novembre - Copy.txt
Size: 50, 2, TSK:/Users/UserX/AppData/Local/Google/Drive/snapshot.db, 48581, -,
sqlite, plugin: google_drive
11/07/2014, 13:22:36, UTC, M..., LOG, Google Drive (local entry), Content Modification
Time, -, WIN-I51P7DIKOO0, %local_sync_root%/Archives photos/photo2.bmp, File Path:
%local_sync_root%/Archives photos/photo2.bmp Size: 1555254, 2,
TSK:/Users/UserX/AppData/Local/Google/Drive/snapshot.db,48581, -, sqlite, plugin:
google_drive
```

Listing D.9: Information extracted from Google Drive using Plaso

Each entry contains the following information:

- *date*, *time* and *timezone*: this information indicates the date on which the file was transferred to the cloud or downloaded to the local machine.
- *MACB* and *type*: information about the type of action. These two fields are set to "M..." and "Content Modification Time".
- *source* and *sourcetype*: type of event. These values are set to "LOG" and "Google Drive (local entry)" or "Google Drive (cloud entry)" depending on the type of operation (upload or download).
- *host*: information about the host (i.e. the machine on which the event happens) of the event.

The *desc* field for local entries is made of the following information: the path of the file and its size. For the extraction of this information, the extraction pattern above is used:

Extraction pattern ¹². *File Path: {filePath} Size: {fileSize}*

The *desc* field for cloud entries is made of the following information: the path of the downloaded file, a flag indicating if the file is shared or not, its size, its URL and its type. For the extraction of this information, the extraction pattern above is used:

Extraction pattern ¹³. *File Path: {filePath} [{isShared}] Size: {size} URL: {url} Type: {documentType}*

D.9/ LINK

The files related to Windows shortcuts (.lnk extension) are used in Windows to access quickly a given file. The formatting of data related to this source is made by the script winlnk.py of Plaso. The entries that can be generated using this script are shown in Listing D.10.

```
date, time, timezone, MACB, source, sourcetype, type, user, host, short, desc, version,
filename, inode, notes, format, extra
11/10/2014, 08:44:32, UTC, .A.., LNK, Windows Shortcut, Last Access Time, -,
WIN-I51P7DIKOO0, [Lancer Skype] C:\Program Files\Skype\Phone\Skype.exe, [Lancer
Skype] File size: 30526056 File attribute flags: 0x00000021 Drive type: 3 Drive
serial number: 0x724766a7 Local path: C:\Program Files\Skype\Phone\Skype.exe
Relative path: ..\..\..\..\..\Program Files\Skype\Phone\Skype.exe Working dir:
C:\Program Files\Skype\ Icon location:
%SystemRoot%\Installer\{24991BA0-F0EE-44AD-9CC8-5EC50AECF6B7}\SkypeIcon.exe, 2,
TSK:/ProgramData/Microsoft/Windows/Start Menu/Programs/Skype/Skype.lnk, 49451, -,
lnk, linked_path: C:\Program Files\Skype\Phone\Skype.exe file_attribute_flags: 33
drive_serial_number: 1917281959
11/10/2014, 08:45:11, UTC, ...B, LNK, Windows Shortcut, Creation Time, -,
WIN-I51P7DIKOO0, [Empty description] C:\Users\UserX\Google Drive\Images, [Empty
description] File size: 0 File attribute flags: 0x00000010 Drive type: 3 Drive
serial number: 0x724766a7 Local path: C:\Users\UserX\Google Drive\Images Relative
path: ..\..\..\..\..\Google Drive\Images, 2,
TSK:/Users/UserX/AppData/Roaming/Microsoft/Windows/Recent/Images.lnk, 49810, -,
lnk, linked_path: C:\Users\UserX\Google Drive\Images file_attribute_flags: 16
drive_serial_number: 1917281959
11/10/2014, 08:45:23, UTC, M..., LNK, Windows Shortcut, Content Modification Time, -,
WIN-I51P7DIKOO0, [Empty description] C:\Users\UserX\Google
Drive\Images\photoIdentite.bmp, [Empty description] File size: 0 File attribute
flags: 0x00000020 Drive type: 3 Drive serial number: 0x724766a7 Local path:
C:\Users\UserX\Google Drive\Images\photoIdentite.bmp Relative path:
..\..\..\..\..\Google Drive\Images\photoIdentite.bmp Working dir:
C:\Users\UserX\Google Drive\Images, 2,
TSK:/Users/UserX/AppData/Roaming/Microsoft/Windows/Recent/photoIdentite.lnk, 49809,
-, lnk, linked_path: C:\Users\UserX\Google Drive\Images\photoIdentite.bmp
file_attribute_flags: 32 drive_serial_number: 1917281959
```

Listing D.10: Information related to the links extracted using Plaso

Each entry contains the following information:

- *date*, *time* and *timezone*: this information indicates the date on which the shortcut was created, modified or used.
- *MACB* and *type*: information about the type of action. These two fields are set to "M..." and "Content Modification Time" if the action is the modification of a shortcut. They are set to "...B" and "Creation Time" if the action is the creation of a shortcut. They are set to ".A.." and "Last Access Time" if the action is the use of a shortcut.

- *source* and *sourcetype*: type of event. These values are set to "LNK" and "Windows Shortcut".
- *host*: information about the host (i.e. the machine on which the event happens) of the event.

The *desc* field contains many information depending on the type of action performed on the shortcut: the size and the attributes of the file related to the shortcut, the absolute and relative path to the file pointed by the shortcut and information about the icon used for the shortcut. For the extraction of this information, the extraction pattern above is used:

Extraction pattern ¹⁴ (*{description}*). *File size: {fileSize} File attribute flags: 0x{fileAttributes} Drive type: {driveType} Drive serial number: 0x{driveSerialNumber} Volume label: {volumeLabel} Local path: {localPath} Network path: {networkPath} cmd arguments: {commandParameters} env location {environment} Relative path: {relativePath} Working dir: {workingDirectory} Icon location: {iconLocation} Link target: {linkTarget}*

D.10/ RECYCLE BIN

The recycle bin used in Windows contains the files deleted by the user. This location is a step towards the final removal of files using the trash dump. The formatting of data related to this source is done by the script *recycler.py* of Plaso. The entries that can be generated using this script are shown in Listing D.11.

```
date, time, timezone, MACB, source, sourcetype, type, user, host, short, desc, version,
filename, inode, notes, format, extra
11/10/2014, 08:37:51, UTC, M..., RECBIN, Recycle Bin, Content Deletion Time, -,
WIN-I51P7DIKOO0, Deleted file: C:\Users\UserX\Desktop\Compte rendu\CR Reunion
Septembre.txt, C:\Users\UserX\Desktop\Compte rendu\CR Reunion Septembre.txt, 2,
TSK:/$Recycle.Bin/S-1-5-21-2714290424/$IM2HK3V.txt,46049, -, recycle_bin,
file_size: 0
11/10/2014, 08:38:38, UTC, M..., RECBIN, Recycle Bin, Content Deletion Time, -,
WIN-I51P7DIKOO0, Deleted file: C:\Users\UserX\Desktop\Compte rendu\fichiers
ressources\courbe ..., C:\Users\UserX\Desktop\Compte rendu\fichiers
ressources\courbe - Copy.bmp, 2,
TSK:/$Recycle.Bin/S-1-5-21-2714290424/$IAU39MT.bmp, 47581, -, recycle_bin,
file_size: 848294
11/10/2014, 08:40:02, UTC, M..., RECBIN, Recycle Bin, Content Deletion Time, -,
WIN-I51P7DIKOO0, Deleted file: C:\Users\Public\Desktop\Safari.Ink,
C:\Users\Public\Desktop\Safari.Ink, 2,
TSK:/$Recycle.Bin/S-1-5-21-2714290424/$ICJOHEI.Ink, 45936, -, recycle_bin,
file_size: 2495
```

Listing D.11: Information extracted from the recycle bin using Plaso

Each entry contains the following information:

- *date*, *time* and *timezone*: this information indicates the date on which the file was put in the recycle bin.

- *MACB* and *type*: information about the type of action. These two fields are set to "M..." and "Content Deletion Time".
- *source* and *sourcetype*: type of event. These values are set to "RECBIN" and "Recycle Bin".
- *host*: information about the host (i.e. the machine on which the event happens) of the event.

The *desc* field contains only the path of the deleted file. This information is extracted using the following pattern:

Extraction pattern ¹⁵. *Deleted file*: {*pathFile*}

D.11/ SKYPE

Skype is a software that allows users to chat via audio calls, visios or instant messaging. Skype also offers features to share files or screens. The analysis of the data produced by Skype can be valuable to investigate the interaction of a given user with potential accomplices for example. Skype stores information about the accounts used on the machine, the text messages sent and received, the calls made and the files shared with other people. The formatting of data related to this source is done by the script *skype.py* of Plaso. The entries that can be generated using this script are shown in Listing D.12.

```
date, time, timezone, MACB, source, sourcetype, type, user, host, short, desc, version,
filename, inode, notes, format, extra
11/10/2014, 08:46:36, UTC, ...., LOG, Skype Chat MSG, Chat from Skype, -,
WIN-I51P7DIKOO0, From: Yoan Chabot <yoan.chabot> To: aurelie.bertaux, From: Yoan
Chabot <yoan.chabot> To: aurelie.bertaux [aurelie.bertaux] Message: [<quote
author="yoan.chabot" authorname="Yoan Chabot" conversation="aurelie.bertaux"
guid="x9d6998d0ccda6ea204658465936e91aa07c212f8913847e94cba817af6ac5350"
timestamp="1415365658"><legacyquote>[11/7/2014 2:07:38 PM] Yoan Chabot:
</legacyquote>Salut ne tiens pas compte de cette discussion <ss
type="wink">);</ss>+<legacyquote>&lt;&lt;&lt;&lt;</legacyquote></quote>], 2,
TSK:/Users/UserX/AppData/Roaming/Skype/yoan.chabot/main.db, 49830, -,
sqlite, plugin: skype
11/10/2014, 08:46:51, UTC, ...., LOG, Skype Call, Call from Skype, -, WIN-I51P7DIKOO0,
From: yoan.chabot To: echo123 [WAITING], From: yoan.chabot To: echo123 [WAITING], 2,
TSK:/Users/UserX/AppData/Roaming/Skype/yoan.chabot/main.db, 49830, -,
sqlite, plugin: skype video_conference: False user_start_call: False
11/10/2014, 08:46:52, UTC, ...., LOG, Skype Call, Call from Skype, -, WIN-I51P7DIKOO0,
From: yoan.chabot To: echo123 [ACCEPTED], From: yoan.chabot To: echo123 [ACCEPTED],
2, TSK:/Users/UserX/AppData/Roaming/Skype/yoan.chabot/main.db, 49830, -, sqlite,
plugin: skype video_conference: False user_start_call: False
```

Listing D.12: Information extracted from Skype using Plaso

Each entry contains the following information:

- *date*, *time* and *timezone*: temporal information.

- *MACB* and *type*: information about the type of action. The *MACB* field is set to "...". The value of the *type* field depends on the type of the action made.
- *source* and *sourcetype*: type of event. The *source* field is set to "LOG". The value of *sourcetype* is set to "Skype Chat MSG", "Skype Call", "Skype Account", "Skype SMS" or "Skype Transfer Files" depending on the type of action.
- *host*: information about the host (i.e. the machine on which the event happens) of the event.

Regarding entries related to instant text messages, the *desc* field is made of the following information: the sender and the receiver of the message, its title and its content. For the extraction of this information, the extraction pattern above is used:

Extraction pattern ¹⁶. *From: {fromAccount} To: {toAccount} [{title}] Message: [{message}]*

The *desc* field of entries related to SMS contains the phone number of the receiver and the content of the message. For the extraction of this information, the extraction pattern above is used:

Extraction pattern ¹⁷. *To: {number} [{message}]*

For a call, the *desc* field contains the pseudonym of the caller, the pseudonym of the person called and the status of the call. For the extraction of this information, the extraction pattern above is used:

Extraction pattern ¹⁸. *From: {from} To: {to} [{status}]*

The *desc* field of entries related to a file transfer is made of the source and the destination path, the name of the file and the type of action needed after receiving the file. For the extraction of this information, the extraction pattern above is used:

Extraction pattern ¹⁹. *Source: {source} Destination: {destination} File: {filename} [{actionType}]*

D.12/ JAVA IDX

Java IDX files are cache files used to enhance the performance of Java Web Start applications. The formatting of data related to this source is done by the script `java.idx.py` of Plaso. The entries that can be generated using this script are shown in Listing D.13.

```
date, time, timezone, MACB, source, sourcetype, type, user, host, short, desc, version,
filename, inode, notes, format, extra
12/02/2014, 13:26:23, UTC, ...B, JAVA.IDX, Java Cache IDX, File Downloaded, -,
WIN-I51P7DIKOO0, IDX Version: 605 Host IP address: (23.206.41.31) Download URL:
http://docs.or...,IDX Version: 605 Host IP address: (23.206.41.31) Download URL:
http://docs.oracle.com/javase/tutorialJWS/ModalityDemo.jnlp, 2,
TSK:/Users/UserX/AppData/LocalLow/Sun/Java/Deployment/ cache/6.0/11/864e.idx,53166,
-, java.idx,
```

 Listing D.13: Information extracted from Java Web Start using Plaso

Each entry contains the following information:

- *date*, *time* and *timezone*: information indicating the date at which the file was downloaded.
- *MACB* and *type*: information about the type of action. The values of these two fields are set to "...B" and "File Downloaded".
- *source* and *sourcetype*: type of event. These values are set to "JAVA_IDX" and "Java Cache IDX".
- *host*: information about the host (i.e. the machine on which the event happens) of the event.

The *desc* field contains the following information: the version of Java Web Start, the IP address of the machine hosting the file and the URL of this file. For the extraction of this information, the extraction pattern above is used:

Extraction pattern ²⁰. *IDX Version: {idxVersion} Host IP address: ({ipHost}) Download URL: {url}*

D.13/ OLE COMPOUND FILE

OLE (i.e. Object Linking and Embedding) is a technology of Microsoft allowing to easily export files (Office files for example) to other applications. The study of these files allows the investigator to get information about the files handled by the user. The formatting of data related to this source is made by the script *olecf.py* of Plaso. The entries that can be generated using this script are shown in Listing D.14.

```
date, time, timezone, MACB, source, sourcetype, type, user, host, short, desc, version,
filename, inode, notes, format, extra
11/06/2014,09:57:20, UTC, ..., OLECF, OLECF Summary Info, Document Creation Time, -,
WIN-I51P7DIKOO0, Title: Installation Database Subject: Skype Author: Skype
Technologies S.A. R..., Title: Installation Database Subject: Skype Author: Skype
Technologies S.A. Keywords: Installer MSI Database Comments: This installer
database contains the logic and data required to install Skype 6.22. Template:
Intel;1033 Revision number: {9B8D3250-AA3C-40D0-9A1F-9A7AD30F5D0D} Number of pages:
200 Number of words: 2 Application: Windows Installer XML (3.0.5419.0) Security: 2,
2, TSK:/ProgramData/Skype/{24991BA0}/Skype.msi;
TSK:/Users/UserX/AppData/Local/Temp/Skype.msi;
TSK:/Windows/Installer/8bb01.msi,48694, -, olecf, name: Summary Information
plugin: olecf-summary
11/06/2014, 09:58:39, UTC, ..., OLECF, OLECF Summary Info, Document Last Save Time, -,
WIN-I51P7DIKOO0, Title: Installation Database Subject: Skype Author: Skype
Technologies S.A. R..., Title: Installation Database Subject: Skype Author: Skype
Technologies S.A. Keywords: Installer MSI Database Comments: This installer
database contains the logic and data required to install Skype 6.22. Template:
Intel;1033 Revision number: {9B8D3250} Number of pages: 200 Number of words: 2
```

```

Application: Windows Installer XML (3.0.5419.0) Security: 2, 2,
TSK:/ProgramData/Skype/{24991BA0}/Skype.msi;
TSK:/Users/UserX/AppData/Local/Temp/Skype.msi; TSK:/Windows/Installer/8bb01.msi,
48694, -, olecf, name: Summary Information plugin: olecf.summary
11/06/2014, 09:58:39, UTC, ...., OLECF, OLECF Summary Info, Document Last Printed Time,
-, WIN-I51P7DIKOO0, Title: Installation Database Subject: Skype Author: Skype
Technologies S.A. R..., Title: Installation Database Subject: Skype Author: Skype
Technologies S.A. Keywords: Installer MSI Database Comments: This installer
database contains the logic and data required to install Skype 6.22. Template:
Intel;1033 Revision number: {9B8D3250} Number of pages: 200 Number of words: 2
Application: Windows Installer XML (3.0.5419.0) Security: 2, 2,
TSK:/ProgramData/Skype/{24991BA0}/Skype.msi;
TSK:/Users/UserX/AppData/Local/Temp/Skype.msi; TSK:/Windows/Installer/8bb01.msi,
48694, -, olecf, name: Summary Information plugin: olecf.summary
11/06/2014, 10:15:35, UTC, M..., OLECF, OLECF Summary Info, Content Modification Time,
-, WIN-I51P7DIKOO0, Title: Installation Database Subject: Skype Author: Skype
Technologies S.A. R..., Title: Installation Database Subject: Skype Author: Skype
Technologies S.A. Keywords: Installer MSI Database Comments: This installer
database contains the logic and data required to install Skype 6.22. Template:
Intel;1033 Revision number: {9B8D3250} Number of pages: 200 Number of words: 2
Application: Windows Installer XML (3.0.5419.0) Security: 2, 2,
TSK:/ProgramData/Skype/{24991BA0}/Skype.msi;
TSK:/Users/UserX/AppData/Local/Temp/Skype.msi; TSK:/Windows/Installer/8bb01.msi,
48694, -, olecf, name: Summary Information plugin: olecf.summary

```

Listing D.14: Information related to OLECF documents extracted using Plaso

Each entry contains the following information:

- *date*, *time* and *timezone*: information indicating the date at which the OLECF file was created, displayed, modified or saved.
- *MACB* and *type*: information about the type of action. The values of these two fields are set to "...." and "Document Creation Time" for the creation of a file. They are set to "...." and "Document Last Save Time" for the save of a file. They are set to "...." and "Document Last Printed Time" for the display of a file. They are set to "M..." and "Content Modification Time" for the modification of a file.
- *source* and *sourcetype*: type of event. These values are set to "OLECF" and "OLECF Summary Info".
- *host*: information about the host (i.e. the machine on which the event happens) of the event.

The *desc* field related to these entries contains the following information: the title of the document and its author, its topic, its keywords, its comments, the template used to create the document, the revision number, the number of pages, the number of words and the number of characters, the application used to edit the document and security information. To extract this information, the following pattern is used:

Extraction pattern ²¹. *Title: {title} Subject: {subject} Author: {author} Keywords: {keywords} Comments: {comments} Template: {template} Revision number: {revision Number} {Last saved by:} {lastSavedBy} Total edit time: {totalEditTime} Number of*

pages: {*numberPages*} *Number of words:* {*numberWords*} *Number of characters:*
{*numberCharacters*} *Application:* {*application*} *Security:* {*security*}

BIBLIOGRAPHY

- Abbott, J., Bell, J., Clark, A., De Vel, O., and Mohay, G. (2006). Automated recognition of event scenarios for digital forensics. In *Proceedings of the 2006 ACM symposium on Applied computing*, pages 293–300. ACM.
- Agarwal, A., Gupta, M., and Gupta, S. (2011). Systematic digital forensic investigation model. *International Journal of Computer Science and Security (IJCSS)*, 5(1):118.
- Alharbi, S. A., Weber-Jahnke, J., and Traore, I. (2011). The proactive and reactive digital forensics investigation process: A systematic literature review. *International Journal of Security and Its Applications*, 5(4):59–72.
- Allen, J. F. (1983). Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843.
- Anderson, R., Barton, C., Böhme, R., Clayton, R., van Eeten, M. J., Levi, M., Moore, T., and Savage, S. (2012). Measuring the cost of cybercrime. In *11th Workshop on the Economics of Information Security (June 2012)*.
- Barnum, S. (2011). Cyber observable expression (cybox) use cases.
- Baryamureeba, V. and Tushabe, F. (2004). The enhanced digital investigation process model. In *Proceedings of the Fourth Digital Forensic Research Workshop*. Citeseer.
- Beebe, N. L. and Clark, J. G. (2005). A hierarchical, objectives-based framework for the digital investigations process. *Digital Investigation*, 2(2):147–167.
- Buchholz, F. and Falk, C. (2005). Design and implementation of zeitline: a forensic timeline editor. In *Digital forensic research workshop*.
- Carbone, R. and Bean, C. (2011). Generating computer forensic super-timelines under linux.
- Carrier, B. et al. (2003a). Defining digital forensic examination and analysis tools using abstraction layers. *International Journal of digital evidence*, 1(4):1–12.
- Carrier, B. and Spafford, E. H. (2004a). An event-based digital forensic investigation framework. In *Digital forensic research workshop*.
- Carrier, B., Spafford, E. H., et al. (2003b). Getting physical with the digital investigation process. *International Journal of Digital Evidence*, 2(2):1–20.

- Carrier, B. D. and Spafford, E. H. (2004b). Defining event reconstruction of digital crime scenes. *Journal of Forensic Sciences*, 49(6):1291.
- Carvey, H. (2009). Timeline analysis, pt iii.
- Case, A., Cristina, A., Marziale, L., Richard, G. G., and Roussev, V. (2008). Face: Automated digital evidence discovery and correlation. *digital investigation*, 5:S65–S75.
- Casey, E., Back, G., and Barnum, S. (2015). Leveraging cybox™ to standardize representation and exchange of digital forensic information. *Digital Investigation*, 12, Supplement 1(0):S102 – S110. {DFRWS} 2015 Europe Proceedings of the Second Annual {DFRWS} Europe.
- Chabot, Y. (2012). Web s'émantique. Technical report, CheckSem Team, Electronics Laboratory, Computer Imaging - UMR CNRS 6306.
- Chabot, Y., Bertaux, A., Nicolle, C., and Kechadi, T. (2014a). Automatic timeline construction and analysis for computer forensics purposes. In *IEEE Joint Intelligence & Security Informatics Conference 2014 (IEEE JISIC2014)*.
- Chabot, Y., Bertaux, A., Nicolle, C., and Kechadi, T. (2014b). A complete formalized knowledge representation model for advanced digital forensics timeline analysis. *Digital Investigation (Fourteenth Annual DFRWS Conference)*, 11(2):S95–S105.
- Chabot, Y., Bertaux, A., Nicolle, C., and Kechadi, T. (2014c). Event reconstruction: A state of the art. *Handbook of Research on Digital Crime, Cyberspace Security, and Information Assurance*.
- Chabot, Y., Bertaux, A., Nicolle, C., and Kechadi, T. (2014d). Reconstruction et analyse sémantique de chronologies cybercriminelles. In *Revue des Nouvelles Technologies de l'Information*, pages 521–524.
- Chabot, Y., Bertaux, A., Nicolle, C., and Kechadi, T. (2015a). De la scène de crime aux connaissances: représentation d'évènements et peuplement d'ontologie appliqués au domaine de la criminalistique informatique. In *Extraction et Gestion des Connaissances 2015*.
- Chabot, Y., Bertaux, A., Nicolle, C., and Kechadi, T. (2015b). An ontology-based approach for the reconstruction and analysis of digital incidents timelines. *Digital Investigation*.
- Chen, K., Clark, A., De Vel, O., and Mohay, G. (2003). Ecf-event correlation for forensics. In *First Australian Computer Network and Information Forensics Conference*, pages 1–10, Perth, Australia. Edith Cowan University.
- Ciardhuáin, S. (2004). An extended model of cybercrime investigations. *International Journal of Digital Evidence*, 3(1):1–22.

- Cram, D., Jouvin, D., and Mille, A. (2007). Visualizing Interaction Traces to improve Reflexivity in Synchronou Collaborative e-Learning Activities. In Limited, A. C., editor, *6th European Conference on e-Learning*, pages 147–158.
- Debar, H., Becker, M., and Siboni, D. (1992). A neural network component for an intrusion detection system. In *Research in Security and Privacy, 1992. Proceedings., 1992 IEEE Computer Society Symposium on*, pages 240–250. IEEE.
- Depren, O., Topallar, M., Anarim, E., and Ciliz, M. K. (2005). An intelligent intrusion detection system (ids) for anomaly and misuse detection in computer networks. *Expert systems with Applications*, 29(4):713–722.
- Ehrig, M., Haase, P., Hefke, M., and Stojanovic, N. (2005). Similarity for ontologies-a comprehensive framework. *ECIS 2005 Proceedings*, page 127.
- Emani, C. K., Cullot, N., and Nicolle, C. (2015). Understandable big data: A survey. *Computer Science Review*.
- Farmer, D. and Venema, W. (2004). The coroner’s toolkit (tct).
- Farrell, M. G. (1993). Daubert v. merrell dow pharmaceuticals, inc.: Epistemology and legal process. *Cardozo L. Rev.*, 15:2183.
- Ferré, S. (2014). Expressive and Scalable Query-Based Faceted Search over SPARQL Endpoints. In *Semantic Web (ISWC)*, pages 438 – 453, Riva del Garda, Italy.
- Forte, D. V. (2004). The art of log correlation: Tools and techniques for correlating events and log files. *Computer Fraud & Security*, 2004(8):15–17.
- Freiling, F. C. and Schwittay, B. (2007). A common process model for incident response and computer forensics. In *Proceedings of Conference on IT Incident Management and IT Forensics*, pages 1–21.
- Ganter, B., Wille, R., and Franzke, C. (1997). *Formal concept analysis: mathematical foundations*. Springer-Verlag New York, Inc.
- Gantz, J. and Reinsel, D. (2012). The digital universe in 2020: Big data, bigger digital shadows, and biggest growth in the far east. *IDC iView: IDC Analyze the Future*, 2007:1–16.
- Garfinkel, S. (2012). Digital forensics {XML} and the {DFXML} toolset. *Digital Investigation*, 8(3–4):161 – 174.
- Gil, Y., Cheney, J., Groth, P., Hartig, O., Miles, S., Moreau, L., Da Silva, P. P., Coppens, S., Garijo, D., and Gomez, J. (2010). Provenance xg final report. *Final Incubator Group Report*.

- Gladyshev, P. (2004). *Formalising event reconstruction in digital investigations: Chapter 2*. PhD thesis, University College Dublin.
- Gladyshev, P. and Patel, A. (2004). Finite state machine approach to digital event reconstruction. *Digital Investigation*, 1(2):130–149.
- Gladyshev, P. and Patel, A. (2005). Formalising event time bounding in digital investigations. *International Journal of Digital Evidence*, 4(2):1–14.
- Gruber, T. R. et al. (1993). A translation approach to portable ontology specifications. *Knowledge acquisition*, 5(2):199–220.
- Gudhjonsson, K. (2010). Mastering the super timeline with log2timeline. *SANS Reading Room*.
- Hargreaves, C. and Patterson, J. (2012). An automated timeline reconstruction approach for digital forensic investigations. *Digital Investigation*, 9:69–79.
- Héon, M. (2014). G-owl: Vers une syntaxe graphique de l'owl humainement lisible "human readable".
- Héon, M. and Nkambou, R. (2013). G-owl: Vers un langage de modélisation graphique, polymorphique et typé pour la construction d'une ontologie dans la notation owl. In *IC-24èmes Journées francophones d'Ingénierie des Connaissances*.
- Hepp, M. (2008). Ontologies: State of the art, business potential, and grand challenges. In Hepp, M., De Leenheer, P., De Moor, A., and Sure, Y., editors, *Ontology Management*, volume 7 of *Computing for Human Experience*, pages 3–22. Springer US.
- Hitzler, P. and Janowicz, K. (2013). Linked data, big data, and the 4th paradigm. *Semantic Web*, 4(3):233–235.
- Ilgun, K., Kemmerer, R., and Porras, P. (1995). State transition analysis: A rule-based intrusion detection approach. *Software Engineering, IEEE Transactions on*, 21(3):181–199.
- Inglot, B., Liu, L., and Antonopoulos, N. (2012). A framework for enhanced timeline analysis in digital forensics. In *Green Computing and Communications (GreenCom), 2012 IEEE International Conference on*, pages 253–256. IEEE.
- James, J., Gladyshev, P., Abdullah, M. T., and Zhu, Y. (2010). Analysis of evidence using formal event reconstruction. *Digital Forensics and Cyber Crime*, pages 85–98.
- James, M., Michael, C., Brad, B., and Jacques, B. (2011). Big data: The next frontier for innovation, competition, and productivity. *The McKinsey Global Institute*.
- Jeyaraman, S. and Atallah, M. J. (2006). An empirical study of automatic event reconstruction systems. *digital investigation*, 3:108–115.

- Karthick, R. R., Hattiwale, V. P., and Ravindran, B. (2012). Adaptive network intrusion detection system using a hybrid approach. In *Communication Systems and Networks (COMSNETS), 2012 Fourth International Conference on*, pages 1–7. IEEE.
- Khan, M. N., Chatwin, C. R., and Young, R. C. (2007). A framework for post-event timeline reconstruction using neural networks. *digital investigation*, 4(3):146–157.
- Kohn, M., Eloff, J., and Olivier, M. (2006). Framework for a digital forensic investigation. In *Proceedings of the ISSA 2006 from Insight to Foresight Conference, Sandton, South Africa (published electronically)*.
- Lebo, T., Sahoo, S., McGuinness, D., Belhajjame, K., Cheney, J., Corsar, D., Garijo, D., Soiland-Reyes, S., Zednik, S., and Zhao, J. (2013). Prov-o: The prov ontology. *W3C Recommendation*, 30.
- Liebig, C., Cilia, M., and Buchmann, A. (1999). Event composition in time-dependent distributed systems. In *Proceedings of the Fourth IECIS International Conference on Cooperative Information Systems*, pages 70–78, Washington, DC, USA. IEEE Computer Society.
- Liew, A. (2007). Understanding data, information, knowledge and their inter-relationships. *Journal of Knowledge Management Practice*, 8(2):1–16.
- Martinez-Cruz, C., Blanco, I. J., and Vila, M. A. (2012). Ontologies versus relational databases: are they so different? a comparison. *Artificial Intelligence Review*, 38(4):271–290.
- Motik, B., Grau, B. C., Horrocks, I., Wu, Z., Fokoue, A., and Lutz, C. (2009). Owl 2 web ontology language: Profiles. *W3C recommendation*, 27:61.
- Ogden, C. K., Richards, I. A., Malinowski, B., and Crookshank, F. G. (1923). *The meaning of meaning*. Kegan Paul London.
- Olsson, J. and Boldt, M. (2009). Computer forensic timeline visualization tool. *Digital Investigation*, 6:78–87.
- Palmer, G. (2001). A road map for digital forensic research. In *First Digital Forensic Research Workshop, Utica, New York*, pages 27–30.
- Perumal, S. (2009). Digital forensic model based on malaysian investigation process. *IJCSNS*, 9(8):38.
- Pilli, E. S., Joshi, R., and Niyogi, R. (2010). Network forensic frameworks: Survey and research challenges. *Digital Investigation*, 7(1):14–27.
- Pollitt, M. (1995). Computer forensics: an approach to evidence in cyberspace. In *Proceedings of the National Information Systems Security Conference*, volume 2, pages 487–491.

- Reith, M., Carr, C., and Gunsch, G. (2002). An examination of digital forensic models. *International Journal of Digital Evidence*, 1(3):1–12.
- Ribaux, O. (2013). Science forensique. <http://www.criminologie.com/article/science-forensique>.
- Richard III, G. and Roussev, V. (2006). Digital forensics tools: the next generation. *Digital Crime and Forensic Science in Cyberspace*. Idea Group Publishing, pages 75–90.
- Robert, M. (2014). Protéger les internautes : Rapport sur la cybercriminalité. Technical report, The French state.
- Rogers, M. K., Goldman, J., Mislan, R., Wedge, T., and Debroya, S. (2006). Computer forensics field triage process model. In *Proceeding of the Conference on Digital Forensics Security and Law*, pages 27–40.
- Schatz, B., Mohay, G., and Clark, A. (2004a). Rich event representation for computer forensics. *Proceedings of the Fifth Asia-Pacific Industrial Engineering and Management Systems Conference (APIEMS 2004)*, 2(12):1–16.
- Schatz, B., Mohay, G., and Clark, A. (2006). A correlation method for establishing provenance of timestamps in digital evidence. *digital investigation*, 3:98–107.
- Schatz, B., Mohay, G. M., and Clark, A. (2004b). Generalising event forensics across multiple domains. In Valli, C., editor, *2nd Australian Computer Networks Information and Forensics Conference*, pages 136–144, Perth, Australia. School of Computer Networks Information and Forensics Conference, Edith Cowan University.
- Schulze-Kremer, S. (1998). Ontologies for molecular biology. In *Proceedings of the Third Pacific Symposium on Biocomputing*, volume 3, pages 695–706. AAAI Press.
- Shin, Y.-D. (2008). New digital forensics investigation procedure model. In *Networked Computing and Advanced Information Management, 2008. NCM'08. Fourth International Conference on*, volume 1, pages 528–531. IEEE.
- Stephenson, P. (2003). A comprehensive approach to digital incident investigation. *Information Security Technical Report*, 8(2):42–54.
- Vanlande, R., Nicolle, C., and Cruz, C. (2008). Ifc and building lifecycle management. *Automation in Construction*, 18(1):70–78.
- Vlachopoulos, K., Magkos, E., and Chrissikopoulos, V. (2013). A model for hybrid evidence investigation. *Emerging Digital Forensics Applications for Crime Detection, Prevention, and Security*, page 150.
- Yusoff, Y., Ismail, R., and Hassan, Z. (2011). Common phases of computer forensics investigation models. *International Journal of Computer Science & Information Technology*, 3(3).

Zikopoulos, P., Eaton, C., et al. (2011). *Understanding big data: Analytics for enterprise class hadoop and streaming data*. McGraw-Hill Osborne Media.

Abstract:

Having a clear view of events that occurred over time is a difficult objective to achieve in digital investigations (DI). Event reconstruction, which allows investigators to build and to understand the timeline of an incident, is one of the most important steps of a DI process. The complete understanding of an incident and its circumstances requires on the one hand to associate each piece of information to its meaning, and on the other hand to identify semantic relationships between these fragments. This complex task requires the exploration of a large and heterogeneous amount of information found on the crime scene. Therefore, investigators encounter cognitive overload problems when processing this data, causing them to make mistakes or omit information that could have a high added value for the progress of the investigation. In addition, any result produced by the reconstruction process must meet several legal requirements to be admissible at trial, including the ability to explain how the results were produced. To help the investigators to deal with these problems, this thesis introduces a semantic-based approach called SADFC. The main objective of this approach is to provide investigators with tools to help them find the meaning of the entities composing the crime scene and understand the relationships linking these entities, while respecting the legal requirements. To achieve this goal, SADFC is composed of two elements. First, SADFC is based on theoretical foundations, ensuring the credibility of the results produced by the tools via a formal and rigorous definition of the processes used. This approach then proposes an architecture centered on an ontology to model and structure the knowledge inherent to an incident and to assist the investigator in the analysis of this knowledge. The relevance and the effectiveness of this architecture are demonstrated through a case study describing a fictitious investigation.

Keywords: Digital forensics, Event reconstruction, Forensic ontology, Timeline analysis

Résumé :

Obtenir une vision précise des événements survenus durant un incident est un objectif difficile à atteindre lors d'enquêtes de criminalistique informatique. Le problème de la reconstruction d'événements, ayant pour objectif la construction et la compréhension d'une chronologie décrivant un incident, est l'une des étapes les plus importantes du processus d'investigation. La caractérisation et la compréhension complète d'un incident nécessite d'une part d'associer à chaque fragment d'information sa signification passée, puis d'établir des liens sémantiques entre ces fragments. Ces tâches nécessitent l'exploration de grands volumes de données hétérogènes trouvés dans la scène de crime. Face à ces masses d'informations, les enquêteurs rencontrent des problèmes de surcharge cognitive les amenant à commettre des erreurs ou à omettre des informations pouvant avoir une forte valeur ajoutée pour les progrès de l'enquête. De plus, tout résultat produit au terme de la reconstruction d'événements doit respecter un certain nombre de critères afin de pouvoir être utilisé lors du procès. Les enquêteurs doivent notamment être en capacité d'expliquer les résultats produits. Afin d'aider les enquêteurs face à ces problèmes, cette thèse introduit l'approche SADFC. L'objectif principal de cette approche est de fournir aux enquêteurs des outils les aidant à restituer la sémantique des entités composant la scène de crime et à comprendre les relations liant ces entités tout en respectant les contraintes juridiques. Pour atteindre cet objectif, SADFC est composé de deux éléments. Tout d'abord, SADFC s'appuie sur des fondations théoriques garantissant la crédibilité des résultats produits par les outils via une définition formelle et rigoureuse des processus utilisés. Cette approche propose ensuite une architecture centrée sur une ontologie pour modéliser les connaissances inhérentes à la scène de crime et assister l'enquêteur dans l'analyse de ces connaissances. La pertinence et l'efficacité de ces outils sont démontrées au travers d'une étude relatant un cas d'investigation fictive.

Mots-clés : Criminalistique informatique, Reconstruction d'événements, Ontologie, Analyse de chronologies

