CLOUD ACCOUNTABILITY PROJECT

D:D-2.2: High-level architecture

Deliverable Number	D42.2
Work Package	WP 42
Version	Final (Version 1.1)
Deliverable Lead Organisation	HP
Dissemination Level	PU
Contractual Date of Delivery (release)	30/09/2014
Date of Delivery	02/10/2014
	14/11/2014 updated version

Editor

Frederic Gittler (HP), Theo Koulouris (HP)

Contributors

Richard Mark Brown (ATC), Alain Pannetrat (CSA), Jean-Claude Royer (EMN), Mohamed Sellami (EMN), Monir Azraoui (EURECOM), Kaoutar Elkhiyaoui (EURECOM), Melek Önen (EURECOM), Anderson Santana De Oliveira (SAP), Karin Bernsmed (SINTEF), Vasilis Tountopoulos (ATC)

Reviewers

Rehab Alnemr (HP), Christoph Reich (HFU)

Executive Summary

The goal of the A4Cloud Reference Architecture is to provide an abstract but powerful model for designing accountability in modern cloud and future Internet ecosystems. It is as an essential step towards addressing the requirements of the target stakeholders by defining the architecture vision and capabilities and delivering a roadmap to implement such requirements in specific cases, aligned with selected business goals.

This document is a time-driven release, structured to provide a high-level view of the architecture for accountability the A4Cloud project is developing. As such it comprises an intermediate step towards the delivery of the full A4Cloud Reference Architecture (final version due month 42 of the project's run).

The high-level architecture described in this document captures in a succinct manner the main technical areas to be developed in relation to one-another and creates a base for assessing the coherence of the tools. The document presents:

- A methodology for applying, from the organization's perspective, an accountability-driven governance approach;
- A set of accountability-support services proposed to tackle the challenges of extending accountability across complex cloud service provisioning chains;
- A description of the tools which are developed by the project to address specific accountability functions and an early view on their integration.

Table of Contents

1 Introduction

In A4Cloud, we adopt the following definition for the Reference Architecture: "*A Reference Architecture is, in essence, a predefined architectural pattern, or set of patterns, possibly partially or completely instantiated, designed and proven for use in particular business and technical contexts, together with supporting artefacts to enable their use*" [1]. As such, the goal of the A4Cloud Reference Architecture is to provide an abstract but powerful model for designing accountability in modern cloud and future Internet ecosystems. It is as an essential step towards addressing the requirements of the target stakeholders by defining the architecture vision and capabilities and delivering a roadmap to implement such requirements in specific cases, aligned with selected business goals.

The ultimate goal of the A4Cloud Reference Architecture is to offer all possible stakeholders in the cloud service provisioning chain, a guidance on how accountability can be implemented across different contexts in the cloud and Future Internet applications and, share technology-driven best practices and tools for delivering accountability-based solutions to end users.

The high-level architecture described in this document captures in a succinct manner the main technical areas to be developed in relation to one-another and creates a base for assessing the coherence of the tools developed by the project. The document presents:

- A methodology for applying, from the organization's perspective, an accountability-driven governance approach;
- A set of accountability-support services proposed to tackle the challenges of extending accountability across complex cloud service provisioning chains;
- A description of the tools which are developed by the project to address specific accountability functions and an early view on their integration.

The project is providing further guidance to ease the technical integration of the various components in the form of a "guidelines and principles" document available separately [2].

This document is a time-driven release, structured to provide a high-level view of the architecture for accountability the A4Cloud project is developing. As such it comprises an intermediate step towards the delivery of the full A4Cloud Reference Architecture (final version due month 42 of the project's run). The designs reported in this document are preliminary in nature and are subject to evolve.

2 Fundamental Concepts

Accountability in the context of handling personal and business confidential information is an important but complex notion that encompasses the obligation to act as a responsible steward of the personal information of others; to take responsibility for the protection and appropriate use of that information beyond mere legal requirements; to be transparent (give account) about how this has been done and to provide remediation and redress. The perceived lack of transparency and control over data governance, inherent in complex cloud service provision chains makes accountability a key market enabler which can help overcome barriers to cloud service adoption. Still, providing accountability both legally and technically in the cloud has proved to be very challenging.

In the A4Cloud project we propose a co-designed approach for accountability that combines a range of technological enhancements with legal, regulatory and governance mechanisms to provide the necessary basis for initiating and sustaining trustworthy data processing and a trusted relationship between data subjects, regulators and information and communications technology (ICT) providers.

Although the goal behind the A4Cloud Reference Architecture (the "RA" hereafter) is to propose a blueprint for end-to-end accountability across the entire cloud service provisioning chain, the starting point for a lot of the concepts and mechanisms discussed focus on making a single organization accountable. The rationale is that the problem of creating accountable cloud supply chains becomes much more tractable if the actors chained together are accountable. Therefore, we begin by defining what it means for an organization to be accountable. In this context, the A4Cloud project has developed a definition of "Accountability for Data Stewardship in the Cloud" and a corresponding model of how various elements of accountability, can be combined to create a roadmap for accountability [3].

As illustrated in Figure 1, the A4Cloud accountability model consists of:

- **Accountability attributes:** central elements of accountability (i.e. the conceptual basis, and related taxonomic analysis of accountability for data stewardship in the cloud). Namely, these are observability, verifiability, attributability, transparency, responsibility, liability and remediability.
- **Accountability practices:** emergent behaviour characterizing accountable organizations (that is, how organizations can incorporate accountability into their business practices). Specifically, an accountable organization:
    - Defines governance to responsibly comply with internal and external criteria, particularly relating to treatment of personal data and/or confidential data.
    - Ensures implementation of appropriate actions.
    - Explains and justifies those actions, namely, demonstrates regulatory compliance that stakeholders' expectations have been met and that organizational policies have been followed.
    - Remedies any failure to act properly, for example, notifies the affected data subjects or organizations, and/or provides redress to affected data subjects or organizations, even in global situations where multiple cloud service providers are involved.
- **Accountability mechanisms:** operational processes, non-technical mechanisms and technical tools that support accountability practices. Operational processes operate at the organizational business process level, by extending existing processes like auditing and risk assessment to support accountability practices. Non-technical mechanisms consist of accountability-reinforcing mechanisms that are predominantly non-technical, such as contracts, policies, codes of conduct, and various legal safeguards and deterrents. Finally, technical tools comprise the various software systems and components an organization may use to carry our various accountability-related operations.

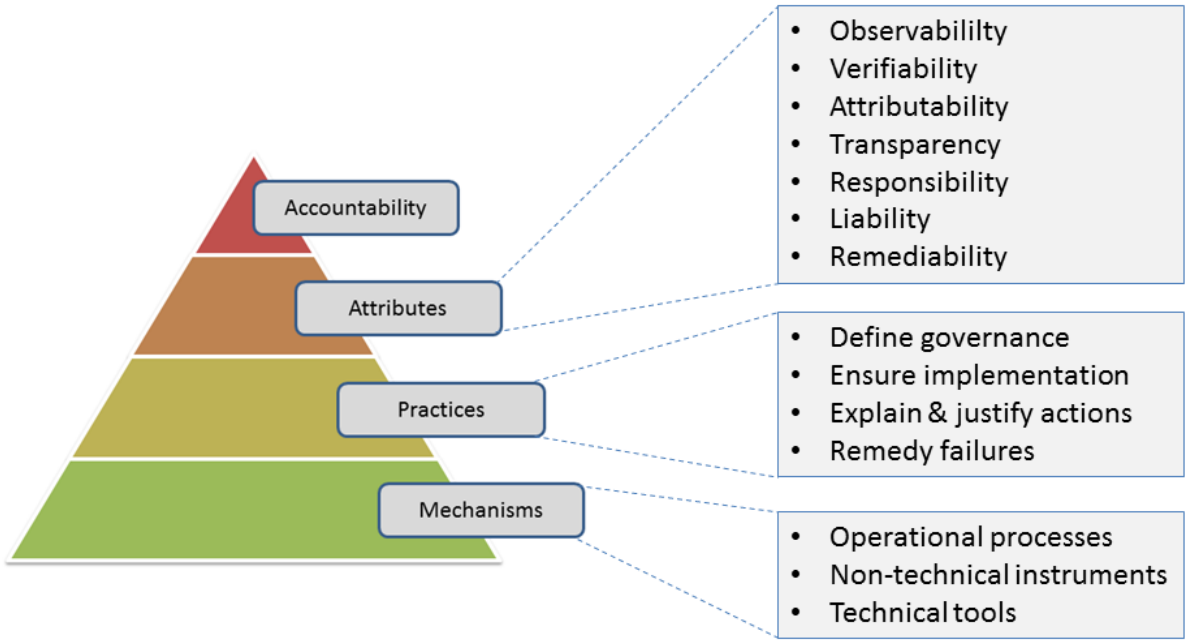

Figure 1: The A4Cloud accountability model.

We may further classify the accountability mechanisms into three categories:

1. Innovative mechanisms designed and built for purpose by A4Cloud (i.e. *"things we build"*).
2. External mechanisms which are imported/utilized by A4Cloud mechanisms (i.e. *"things we import"*).
3. External mechanisms with which A4Cloud mechanisms will co-exist (i.e. *"things we interface with"*).

Table 1 contains an initial categorization of representative mechanisms based on this taxonomy. Although, in developing the reference architecture we will investigate the full spectrum of mechanisms, for the purpose of this document we will focus on the A4Cloud toolkit, i.e. the intersection of "technical tools" and "things we build", presented in Chapter 5.

	Operational processes	Non-technical instruments	Technical tools
Things we build	• Lifecycle for accountability	• Contracts • Policies • Codes of conduct	• A4Cloud Tools
Things we import	• Auditing • Risk assessment	• Certification • Insurance	• Data transformation • Data encryption • Data storage • Anonymization • Sticky policies
Things we interface with	• Existing business lifecycle processes	• Litigation • Sanctions	• Cloud infrastructures & technologies

Table 1: Accountability mechanism taxonomy.

A core element of the concept of accountability is the "account". Within an accountable system, the "account" can be seen as a demonstration of the system's behaviour. We identify three types of "account": proactive account, account of legitimate event(s) and account of incident(s) (see Section 3.9

for details). The description of an “account”-related event provides answers to the six “reporters’ questions”:

- Who: identifies actors involved in the described event.
- What: describes what the account is about.
- Where: describes where the event related to the account occurs (not only physical location).
- When: depicts when the described event occurs.
- Why: presents why the event happened (to respect policies/obligations for instance).
- How: illustrates the used means (logs, encryption, etc.) for the described event.

An “account” also comes with evidence, when possible, associated to these different answers and means for remediation if adequate (the case of an account on an incident for instance).

2.1 Actors and Roles

A key challenge when reasoning about accountability in a cloud context is the adoption of a common vocabulary for expressing fully and consistently elements coming from both the world of technology and from the domain of data protection. The need for a common vocabulary is particularly relevant for the definition of actors and roles in the A4Cloud RA.

The NIST Cloud Computing Reference Architecture [4] defines five major actors:

- Cloud Consumer: “A person or organization that maintains a business relationship with, and uses service from, Cloud Providers.”
- Cloud Provider: “A person, organization, or entity responsible for making a service available to interested parties.”
- Cloud Auditor: “A party that can conduct independent assessment of cloud services, information system operations, performance and security of the cloud implementation.”
- Cloud Carrier: “An intermediary that provides connectivity and transport of cloud services from Cloud Providers to Cloud Consumers.”
- Cloud Broker: “An entity that manages the use, performance and delivery of cloud services, and negotiates relationships between Cloud Providers and Cloud Consumers.”

Although these five roles are sufficient for representing the vast majority of interactions involved in a cloud service provision and procurement context, they do not effectively capture all the elements necessary to reason about accountability. Specifically, as it is, the NIST model cannot capture the following two roles:

- Data “owners”: Individuals, in particular data subjects or organizations who have some personal or confidential data processed in the cloud, and who may not necessarily be qualified as ‘cloud customers’ (or consumers) in the NIST taxonomy. Though more rarely, this also applies to businesses, which may have business confidential data processed by the cloud despite not being a cloud customer (rather customers of a cloud customer). They are essentially “invisible” in the NIST model, but represent the ultimate role in an accountability chain.
- Supervisory authorities: Data protection authorities or telecom regulators may be seen as auditors, but they also have the distinct characteristic of holding enforcement powers, which auditors lack.

In the interest of maintaining maximum compatibility and alignment with the NIST model which appears to be well understood amongst cloud stakeholders, we chose to extend it to cover these roles and support accountability, as follows[1]:

1) Cloud Subject: An entity whose data is processed by a cloud provider, either directly or indirectly. When necessary, we may further distinguish between:
   a) Individual Cloud Subject, when the entity refers to a person.
   b) Organization Cloud Subject, when the entity refers to an organization.
2) Cloud Customer: An entity that (1) maintains a business relationship with, and (2) uses services from a Cloud Provider. When necessary we may further distinguish between:
   a) Individual Cloud Customer, when the entity refers to a person.
   b) Organization Cloud Customer, when the entity refers to an organization.

[1] For an extended discussion please refer to the A4Cloud Conceptual Framework document [3].

3) Cloud Provider: An entity responsible for making a (cloud) service available to Cloud Customers
4) Cloud Carrier: The intermediary entity that provides connectivity and transport of cloud services between Cloud Providers and Cloud Customers
5) Cloud Broker: An entity that manages the use, performance and delivery of cloud services, and negotiates relationships between Cloud Providers and Cloud Customers
6) Cloud Auditor: "An entity that can conduct independent assessment of cloud services, information system operations, performance and security of the cloud implementation, with regards to a set of requirements, which may include security, data protection, information system management, regulations and ethics.
7) Cloud Supervisory Authority: An entity that oversees and enforces the application of a set of rules.

These roles both share similarities and articulate differences with the cloud computing roles defined in the NIST model in the following way:

- We introduce a new role (Cloud Subject) to designate an entity that owns data, which is either directly transferred to a cloud provider for processing, or indirectly through a cloud customer. We further distinguish Cloud Subjects as individuals or organizations.
- The role of Cloud Customer is aligned with the NIST definition (as a synonym of Cloud Consumer) but we further introduce a distinction between individual Cloud Customers and organizational Cloud Customers.
- The roles of Cloud Provider and Cloud Broker are adopted without modification from the definition provided by NIST.
- The role of Cloud Auditor is based on the definition provided by NIST but was altered to better reflect the goals of accountability, by additionally referencing data protection as well as regulatory and ethical requirements.

We note that the role of Cloud Carrier defined by NIST is unlikely to be considered in the context of accountability, since a Cloud Carrier does not normally take any responsibility for data stewardship but merely acts as a neutral transporter (much like an Internet Service Provider). In the case where a Cloud Carrier takes a stronger role in terms of data stewardship, or if the routing of data traffic matters, we may consider it as a Cloud Provider instead without loss of generality.

Even in its extended form, however, the cloud role classification alone cannot provide all the information necessary to fully characterize an actor. For example, a cloud provider can be either a data controller or a data processor, with fundamentally different responsibilities in each case. For that reason, the proposed model for fully specifying an actor's role in the A4Cloud Reference Architecture is to provide both the (extended) cloud and the data protection (95/46/EC and 2002/58/EC) role classifications. Table 2 below presents all the possible combinations of cloud computing and data protection role classifications identified in RA. In conclusion, to fully characterize an actor in the RA and documents produced by the A4Cloud project in general, the proposed nomenclature combining cloud and data protection roles, as presented in Table 2, should be used.

Extended NIST cloud roles	Data protection roles
Cloud subject	Data subject
Cloud customer	Data controller or Data processor
Cloud provider	Data processor or Data controller
Cloud carrier	Data processor or Data controller (unlikely) or Not applicable.
Cloud broker	Data processor or Data controller

Cloud auditor	(Not Applicable)
Cloud supervisory authority	Supervisory authority (DPA or NRA)
(Not Applicable)	Third party
(Not Applicable)	Recipient

Table 2: A4Cloud Reference Architecture roles.

2.2 Conceptual Model of the Reference Architecture

With the essential actors and roles necessary to describe accountability relationships identified, the building blocks of the RA model can now be examined. Again, emphasizing the importance of building upon established and standardized concepts instead of "re-inventing the wheel" we base the RA conceptual model on the corresponding NIST cloud reference architecture conceptual model [4]. Figure 2 below illustrates how the RA adapts and extends the NIST cloud reference architecture to support accountability, by adding new actors as well as "accountability" as a cross-cutting concern alongside security and privacy, since in order to be effective, all three need to be implemented across all layers and functions.

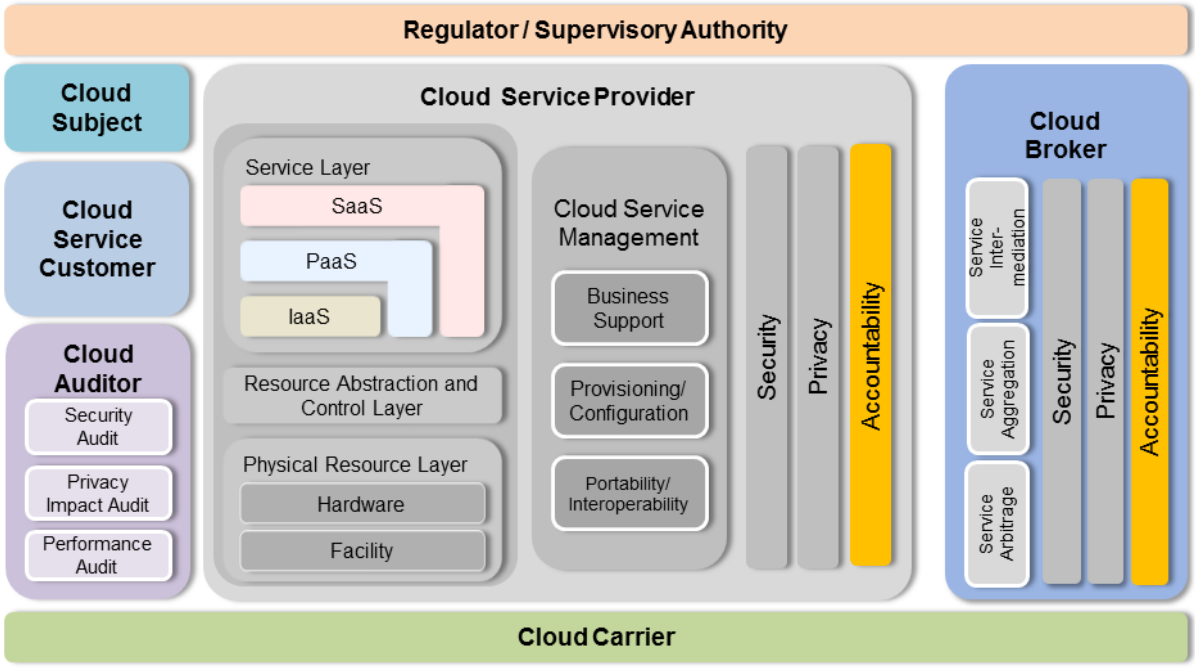

Figure 2: A4Cloud Reference Architecture conceptual model.

3 Accountability Governance

Operating in an accountable manner is not simply a matter of deploying the tools to implement technical controls and to report on their behaviour. It actually starts at the very top of the organization, with the Board of Directors, is embedded in the foundation values of the organization ("organization DNA"), and is transmitted through the whole organization through governance. In the following sections, we will explore the practices required to operate in an accountable manner.

3.1 Accountable Organisations

The Accountability for Cloud Conceptual Framework [3] defines accountable organisations as being *one that takes an accountability-based approach, implying the adoption of the entire set of the accountability practices*. The Conceptual Framework then expands on accountability at an organisational level, focusing on the ways they could be implemented in practice. For the benefit of the reader, these conclusions are listed below:

The Galway project [5] *has defined the central elements that an accountable organisation needs to address as being:*

1. *Organisation commitment to accountability and adoption of internal policies consistent with external criteria.*
2. *Mechanisms to put privacy policies into effect, including tools, training and education.*
3. *Systems for internal, ongoing oversight and assurance reviews and external verification.*
4. *Transparency and mechanisms for individual participation.*
5. *Means for remediation and external enforcement.*

Influenced by this approach, the Canadian privacy commissioners have specified the measures that an accountability management program (for the data protection domain) would ideally include [6]*:*

1. *establishing reporting mechanisms and reflecting these within the organisation's privacy management program controls*
2. *putting in place privacy management program controls, namely:*
   - *a Personal Information Inventory to allow the organisation to identify the personal information in its custody, its sensitivity and the organisation's authority for its collection, usage and disclosure*
   - *policies relating to: collection, use and disclosure of personal information (including requirements for consent and notification); access to and correction of personal information; retention and disposal of personal information; privacy requirements for third parties that handle personal information; security controls and role-based access; handling complaints by individuals about the organisation's personal information handling practices*
   - *risk assessment mechanisms*
   - *training and education*
   - *breach and incident management*
   - *procedures for informing individuals about their privacy rights and the organisation's program controls*
3. *developing an oversight and review plan that describes how the organisation's program controls will be monitored and assessed*
4. *carrying out ongoing assessment and revision of the program controls above*

Furthermore, the proposed EU General Data Protection Regulation (GDPR) [7] *includes many accountability elements including, in Article 22, a list of a Data Controller's accountability instruments:*

- *Policies*
- *Documenting processing operations*
- *Implementing security requirements*
- *Data Protection Impact Assessment*
- *Prior authorisation/consultation by Data Protection Authorities (DPAs)*
- *Data Protection Officer*
- *If proportional, independent internal or external audits*

While we have adopted a "*focus on the data protection domain and on accountability of organizations rather than individuals*" [3], one of our main concern is accountability in the context of IT supply chains based on the use of Cloud Services. End-to-end accountability, which is further analysed in section 4, requires all actors of the supply chain to be accountable organisations to a certain degree. However, the domain for which these organisations need to be accountable is not necessarily the data protection domain. For example, when an organization implements a service which is handling sensitive data (in regards to the data protection regulations) through the use of an IaaS cloud service provider, the latter is typically accountable for providing adequate security, and not for implementing an accountability-based data protection program.

It should not be noticed that, even if the focus as stated above is on organizations, the role of individuals involved is also essential and that accountability must be ensured down to the employee level. Accountable organizations must provide individuals with the necessary tools and procedures to be individually accountable.

In the remainder of this section, we have attempted to define the measures that should be implemented by an accountable organisation in a manner which remains agnostic to the domain. The recommendations have been identified based on work done for both the data protection domain, such as CNIL [8], ICO [9], and Nymity [10]. These have been augmented by more general organizational standards, such as COBIT [11] and ISO 27001 [12]. The author has also leveraged the HP Security Handbook [13] as well as his own professional experience.

3.2 Introduction to the Organisational Lifecycle for Accountability

The Conceptual Framework introduces the Organisational Lifecycle and introduces the Functional Elements of Accountability, which provides the reference model for this discussion.

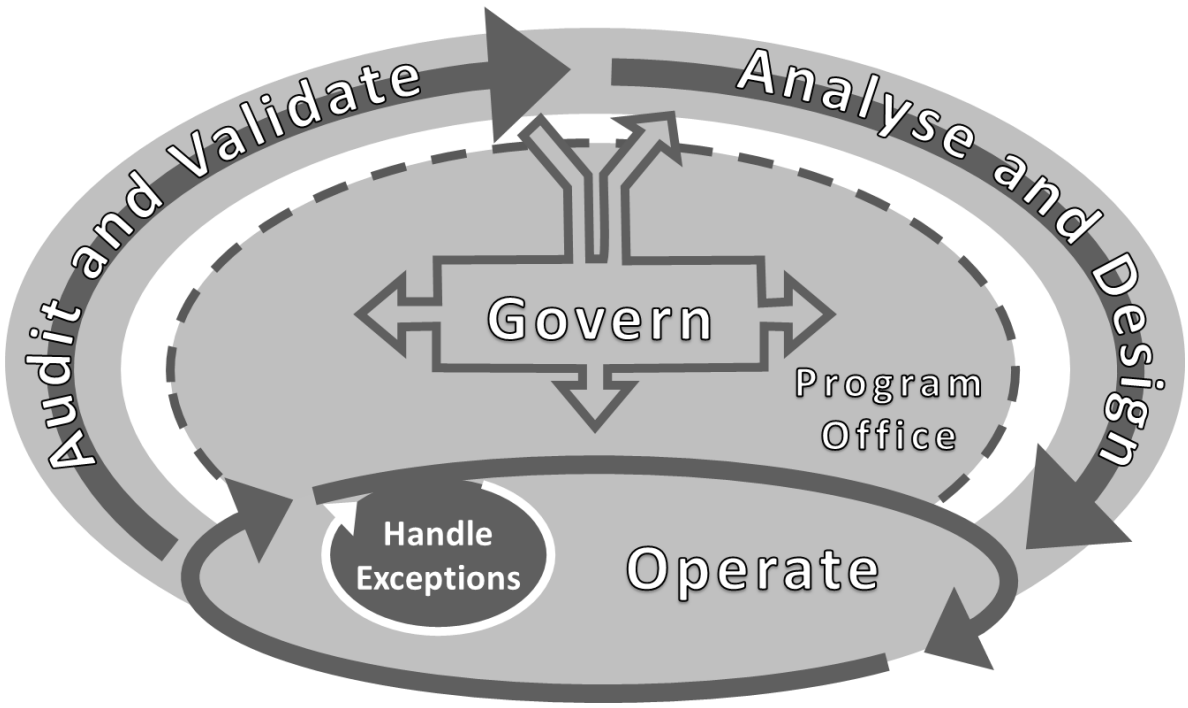

Figure 3: Organisational Lifecycle for Accountability

This lifecycle is organized around five phases which provide a structure to the solution development, operation, and maintenance. Specific to the Corporate Accountability scope, we introduce a sixth element, the Program Office, which provides the operational support to the governance body. Note that the first two elements (Govern and Program Office) are strongly associated with organisational accountability, while the three first lifecycle elements (Analyse and Design, Operate, Handle Exceptions)

describe the lifecycle for building and operating an “accountable solution”. The last element (Audit and Validate) is applicable to both domains, as the assessments can be either focused on the organisation as a whole or on a particular solution or business service.

1. Govern – This corresponds to the executive roles in the organisation establishing and maintaining a framework and supporting management structure and processes, as well as accepting and providing assignment of responsibility, to meet the obligations of the organisation in an accountable manner

2. Program Office – This is the operational body which supports the governance body in meeting its responsibilities in e.g. drafting guidelines, policies and procedures, defining the operational programs and infrastructure, and providing oversight and support for the implementation of the decisions of the governance body. This program office is typically in charge of both organisational accountability as well of the domain for which the organisation is accountable (e.g. the privacy program office or the security program office); it can either be an organisational or a logical structure.

3. Analyse and Design – This corresponds to the analysis and design phases related to the engineering of a solution. The work performed in this phase clearly separates identification of risks (based on business impact, not just technology), identification of controls, design of control implementation, and implementation of controls through technology and processes.

4. Operate – This corresponds to the operational (production) phase of the solution, and includes all the associated management processes

5. Handle Exceptions – This set of activities, which could be considered as an integral part of operations, has been singled-out due to its specific nature and high relevance to accountability. It includes all processes for the handling of complaints and breaches related to accountability obligations

6. Audit and Validate – This corresponds to the assessment of the effectiveness of the controls which have been deployed, the necessary reporting, and paves the way to the tuning (adaptation) of the measures deployed to ensure the obligations are being met.

The following sections describe in more detail the content of each of these phases. More than a general discussion and framing of the scope, we want to provide practical guidelines for implementing accountability. We are providing a series of recommendations which can be used as a checklist. We do not claim this describes a specific methodology but provides a general guideline on integrating accountability within an organization.

These lists are not comprehensive, and each of the points must be evaluated in regards to the size and structure of the organisation. The full list of recommendations may, in general, not be applicable to smaller groups such as SMEs. We acknowledge there are many common points between the recommendations identified below and the actions identified as required for topics like data protection, business continuity, disaster recovery, information security management, and trustworthy accounting. However our recommendations below are not intended to be a substitute for those lists of actions – an organisation must address all of them in order to have a comprehensive coverage and meet its obligations. The analysis focuses on the processes to be deployed by the accountor rather than those of the accountee.

3.3 Roles and Responsibility of Governance Bodies

In the scope of our analysis, we consider obligations which have to be met by the organisation as a whole and where the responsibility for fulfilling these obligations rests with the board members and executive managers, with some part of it going down to the employees. In this context, being responsible often goes beyond the civil responsibility as laws often assign penal sanctions for not meeting obligations and not performing due diligence.

The governance bodies are the owners of the strategic dimension of accountability. In order to fulfil this mission, the board and executive management must:

- Understand relevant obligations in breadth and in depth
- Understand the consequences of not fulfilling the obligations
- Accept responsibility for fulfilling these obligations in an accountable and responsible manner – this is applicable not only to the governance body as a whole, but must be an integral part of the mission of each member of the governance body, which must embrace the obligations, ensure support from each (relevant) functional area, and act as champions in the organisation
- Define the "internal criteria" for the organisation.
- Understand the risks associated with the operation of the business in regards to the obligations. Define a "risk appetite" used as guidance for operational decisions, taking into account the nature of the obligations (ethical, social, or industry norm, contractual, regulatory, legal, etc…). The acceptable level of risk acceptance delegated to the various levels of the organization will typically increase with the management levels in the organization.
- Appoint an executive-level owner who will oversee and be accountable for the fulfilment of the obligations. For example, this is typically the Chief Privacy Officer or the Chief Security Officer for (respectively) the data protection domain or the security domain.
- Ensure the proper integration of all responsibilities and actions across the whole organisation. Avoid operating the program as a "silo" or an afterthought.
- Ensure a proactive attitude towards the accountability domain (e.g. data protection) across the organization.
- Drive the adoption of an accountability-driven mindset. Ensure that this becomes part of the culture, and is integrated with the core values of the organisation (e.g. code of conduct, ethical guidelines, list of values).
- Ensure accountability is properly integrated in all relevant processes (e.g. business management, risk management, compliance management, reporting)
- Ensure that employees are properly trained to understand the concept of Accountability and their own obligations.
- Ensure that Employees are provided with appropriate tools and processes to fulfil their own part of the Accountability obligations.
- Ensure the organisation is ready to respond to discontinuities in compliance to obligations (incident response)
- Regularly review the status of the organization in regards to the compliance to the obligations

In order to fulfil this mission, depending on the scope, the organisation's high level management will typically create a Program Office to provide the operational support for the enactment of the governance decisions and more generally support the governance body in meeting its responsibilities. The Program Office will typically report to the executive-level owner. In relationship with this Program Office, the governance body will:

- Define the mission and charter of the Program Office
- Define the level of authority of the Program Office
- Ensure the Program Office is provided with the necessary means, in terms of resources, personnel, funding and authority so it can fulfil its mission
- Support and champion the various policies, programs, processes and other actions identified by the Program Office as necessary to meet the obligations
- Regularly review the work performed by the Program Office. Use external audits to get an external view on the performance of the Program Office

3.4 **The Program Office**

The program office may be focused on accountability across the organisation, but will more typically be in charge of both the domain (or one of the principal domains) for which the organisation is accountable (e.g. the privacy program office or the security program office) and of accountability. The program office can either be an organisational or a logical structure.

In its role to support the governance bodies in fulfilling their responsibilities, the mission of the Program Office can be divided in eight main areas:

- Inventory Obligations, Risk Assessment, and Risk Treatment
  - Maintain a registry of all obligations and associated operational standards
  - Perform risk assessments and identify risks and exposure. This is focusing on the business and related (accounting, …) practices
  - Identify how to treat the risk. The analysis must include cost, timing, alternatives, and comply with the "risk appetite" identified by the Board.
  - Create a rollout plan
  - Create a set of metrics to report on the state of the accountability program
  - Investigate best practices and compliance frameworks, consider adoption, attestation or certification based on benefits, costs, and risk.

  This sets the stage to perform due-diligence, which often defines the boundary where the responsibility of the officers of the organisation is engaged. Performing due-diligence is however not enough – it can only be used as defence if it can be demonstrated. The Program Office must be sure that this can be done.

- Company Culture, Practices and Standards
  - Review relevant company codes, operating guidelines, and standards in regards to obligations. This must take a holistic view and deal with all appropriate business functions: sales, marketing, business operations, IT, facility management, workplace solutions, finances, accounting…
  - Draft appropriate changes to these codes
  - Rollout these changes and ensure effective change of the documentation throughout the organization
  - Notify staff of changes through adequate communication programs (awareness)
  - When none exist, foster organisation-level codes and standards creation by or in collaboration with businesses & functions when required for alignment of the internal business processes. Ensure compliance through checklists and metrics used for reviews at different levels
  - Build templates for analysis (e.g. impact assessment, risk assessment) in regards to accountability and obligations, for use in the Analyse and Design phase of the lifecycle.
  - Define a sign-off process with responsible parties to validate the major milestones in the Organisational Lifecycle for Accountability

  It is important to adopt the principle that all actions performed be traceable to the persons performing and authorizing it (attributability). This is to be used primarily for root-cause analysis, continuous improvement and individual accountability.

- Incident Recovery and Response

  This is a critical success factor. It is the responsibility of the Program Office to ensure that an adequate structure and set of processes is effectively implemented in the organisation. Considering the scope, this is often best implemented as a pan-organisation structure, rather than smaller teams dedicated to individual product offerings or, at a minimum, that such a structure exists to support dedicated product teams in case of an exceptional event. Details of this program are provided in section 3.7.

- 3rd Party Engagement
  - Ensure, with active involvement of procurement and contract negotiator, that all service and other provisioning contracts are compliant with relevant obligations
  - If appropriate, define appropriate standards and practices in regards to engagement with third-parties
  - Ensure that procurement or other relevant organizations maintain and update a registry of third-party engagements and their relationship with obligations
  - Enforce strict compliance with the standards and practices, as well as reporting
  - Monitor that procurement or other relevant organizations ensure that contract renewal and changes in terms are properly tracked
  - Ensure 3rd party providers are regularly reviewed by procurement or other relevant organizations

- Ensure that processes are in place by procurement or other relevant organizations to deal with non-compliance of 3rd parties

The collection and archiving of contracts is required but not sufficient in most instances. It must be possible to get a quick understanding of the relationship of external engagements with obligations across all engagements (hence the need to maintain a registry). Also refer to the discussion on 3rd parties in section 3.5.

- Employee Skills and Awareness
  - Ensure the organisation maintains a registry of job (function) profiles in relationship with the obligation and identifies "sensitive positions"
  - Ensure the organisation defines recruiting criteria for sensitive positions re. obligations
  - Ensure the organisation has specific training programs for sensitive positions re. obligations
  - Ensure compliance with legally-required training and certification
  - Inject topics in the on-boarding and recurrent employee code of ethics and business training programs
  - Ensure the organisation includes questions measuring accountability awareness and attitude in employee surveys
  - Ensure the organisation organizes and rolls-out specific accountability awareness campaigns, such as posters displayed on billboards and internal bulletins
  - Ensure that management "state of business" (coffee talks) presentations regularly address accountability
  - In more general terms, ensure the organisation addresses accountability in appropriate employee information vehicles with adequate messages for rollout through the organization
  - Ensure the organisation keeps skills of the specialized staff current – support staff to join industry associations, professional networks, specialized conferences, and get training as required

Accountability for one's own actions, regardless of the domain to which it is applied, must become part of the culture of the organisation, must be embraced by all employees and contractors. Circumventing this must be treated as a serious performance issue. It must be noted that human interaction is one of the weakest points in the security of a system.

- Compliance Readiness
  - Appoint liaison to external domestic and regional compliance and regulatory agencies (as appropriate)
  - Appoint liaison to relevant foreign compliance and regulatory (as appropriate)
  - Ensure the organisation tracks "external criteria" – use a mix of specialized information services, industry associations, professional networks, specialized conferences, and consultants.
  - Ensure the organisation understands reporting requirements and ensures compliance
  - Maintain the legally-required documentation (or ensure it is maintained by the relevant departments)

- Deploy Individual Accountability Tools
  - Investigate and (if adequate) ensure deployment of techniques and tools supporting authorization based on duty segregation
  - Investigate and ensure deployment of tools guaranteeing that all actions are logged and allow the identification of the agent and of the authorizer (as appropriate). This must be done in compliance with legal constraints on the handling of individually identifiable information. Ensure that these tools bear a proper timestamp and are secured against tampering or destruction.

There are some commercially-available tools which act as portals and allow the deployment of these types of controls even if the native applications do not support the functionality. In addition, using a uniform mechanism across the organisation allows for a streamlined management of the authorisation structure and of the audit logs. One must note that provisioning the trusted log with attribution is more important, less expensive, and less problematic than deploying the authorization framework due to the complexity of modelling and allocating the correct structure for the roles.

- Ensure continuity of the accountability program
  - Ensure that all the above documentation stays current.
  - Periodically review the various analyses performed
  - Define and maintain a dashboard providing a synthetic view of the accountability program
  - Define and track a set of metrics to measure effectiveness and progress. A significant part of these metrics must correspond to objective (vs. subjective) criteria.
  - Have the accountability program and the Program Office audited regularly (one the order of once a year) by external auditors.

3.5 Provisioning for Accountability – Analyse and Design

By its very nature, accountability identifies and addresses potential risks, harms and expectations – there would be no need for accountability if one could provide a total and absolute guarantee that all these obligations are met. Designing for accountability is therefore naturally associated with the lifecycle dealing with security, data protection, and risk. There are many variants for this lifecycle – we will use the pragmatic model described in [13].

- Understand the product or service and related assets
  - Have a description of the functionality and associated non-functional requirements
  - Inventory the (related) obligations for which the organisation will be accountable
  - Inventory the assets related to the functionality and obligations
  - Perform an impact assessment for these assets in order to qualify the risk
  - Keep and update a record of assets and impacts

- Perform a risk analysis
  - Identify the position of the board and executive management
  - Perform a risk analysis – in regards to accountability, this should focus on obligations and the contributing factors, as well as on the accountability support system. The risk analysis must take in consideration all aspects impacting the organisation, including secondary impacts such as a loss of reputation, product or service implementation, slow down, regulators push back,...
  - Keep and update the record of threats and vulnerabilities, qualified with probability and impact. Associate this record with the record of assets and impacts.

  Risk analysis is a complex process for which many methodologies and software support exist for traditional security related domains. Risk analysis related to accountability is only a part of the overall risk analysis process with a lot less existing tools and practices. At this stage, we make no recommendation on a methodology or set of tools to use but more about this will be discussed in future stages of the project.

- Define the risk treatment
  - Define how risk will be handled as part of the investigation, design and engineering phases for the product or service
  - For each risk, identify if it will be reduced, mitigated, assigned, transferred, or accepted. The residual risk must be understood and treated in a recursive manner until the constraints associated with the obligations are met (typically, the residual risk in the last iteration will be transferred or accepted).
  - Define the controls which will be deployed to reduce or mitigate each risk, and define how the risk will be assigned or transferred, as appropriate. These controls can be based on a mix of technology and processes. In this latter case, a link must be established with the inventory and operational programs defined at the global level for the organisation (see in particular sections 3.4 and 3.7).
  - Define the metrics and dashboard for continuous monitoring of the state and effectiveness of the risk treatment plan.
  - Augment the above record of threats and vulnerabilities with the record of risk treatment and associated measures. Include all steps in the recursive treatment of residual risk.
  - Validate that the risk analysis takes in consideration the selected implementation decisions. Update it as necessary.

- Ensure that the accountability mechanisms are tested as part of the solution testing (in both regression and system tests). Validate the effectiveness of the measures.

This step is tightly integrated with the solution design and engineering phase, and is recursive by nature as each implementation alternative has an impact on risk. The information obtained in this process provides the foundation for signoff and the creation of the first account.

The result of this phase will be the definition of a technical solution to implement the solution or offering. The technical or procedural controls implemented correspond to a due-diligence and best effort coverage of the requirements. It is in general impossible to guarantee that the mechanisms and procedures effectively deployed guarantee that the obligations will be met. The accountability system must be able to handle the unexpected.

- Selection of 3rd party providers

  Special attention must be given the selection of 3rd party providers. We will focus on the selection of cloud services. This section leverages the recommendations made in [14] and [8], but have been considerable modified to address accountability in general as opposed to DPR.
  When selecting a cloud provider, the accountable organization must:
  - Identify assets which will be processed or stored in the cloud environment.
  - Identify the related obligations and associated accountability requirements
  - Based on the initial risk analysis, identify the risk profile of the provider, the set of security (and other relevant attributes) that must be supported
  - Identify the links between the third-party accountability provisions and the accountability system of the organisation – list the associated requirements
  - Based on compliance obligations, identify the certifications and other levels of guarantees that must be offered by the provider (most often including constraints on the local for the data centre and the IT staff)
  - Check internal policies and procedures in terms of selection, registration, and tracking of third party service providers. Comply once the provider is selected.
  - Review and select provider based on a review of the offerings matching the above requirements. Validate the costs are in line with funding expectations. Review the risk appetite and risk treatment plans if there is a gap. The practice of increasing the levels of risk acceptance to meet cost expectation should be banned.
  - Ensure the proper contractual clauses are in place, especially as in regards to compliance to the requirements and the associated accountability measures. The contracts must be handled per organisational policies (see section 3.4)
  - Ensure the provider exploits the data only as intended and defined by the Data Controller
  - Exploitation for the benefit of the provider should be avoided, but if envisaged it must be carefully defined by contract and its consequences (liability and Data Controller role and relevant obligations) must be clearly understood by the organisation and be compliant with the allowable use of the data and legal requirements.
  - Identify the metrics that will be used during the lifetime of the relationship to continuously assess the compliance of the 3rd party.
  - Track changes in the service provider operations.
  - Ensure there is an adequate link between the provider exception handling processes and those of the organisation (see section 3.7). The associated procedures must be tested regularly and readiness assessments must be performed.

- Documentation and Signoff
  - The solution must be fully documented and placed under change management. Any evolution must be assessed against the requirement, obligations, and risks, and necessary adjustments must be performed.
  - The contracts associated with the solution must reflect what is actually implemented
  - An account reflecting the analysis linking obligations with actual controls must be produced and made available to stakeholders. This account is to demonstrate, through a static analysis, the effectiveness of the controls as due-diligence to meet the obligations.
  - The solution must go through a signoff process that will validate that all internal requirements and obligations have been met.

3.6 Operating in an Accountable Manner

This phase covers the support, management, and day-to-day operations. For the purposes of accountability, it focuses on two aspects:

- Operate the system as intended:
  - Gather and report on accountability and risk treatment metrics, keep the dashboards updated
  - Communicate with stakeholders as intended
  - Ensure that the collection of evidence is performed as intended
  - Ensure that all logs are effectively backed-up and are protected against tampering
  - More generally, ensure that all solution-specific processes and associated organisation-level processes are used and are operating with the intended effectiveness. This is also applicable to all processes related to 3rd parties.

- Look for signs of unexpected issues:
  - Continuously monitor the system, the operating environment, and the ecosystem for signs of incident, breach or significant change. Activate the exception handling processes as required (see section 3.7)

3.7 Handling Exceptions

It is critical to have a set of policies and processes to handle exceptions, along with the proper organisation to handle these exceptions quickly and completely. These exceptions can be significant, leading to a workload that cannot promptly be handled by an in-house structure of the organisation. Careful planning and coordination with external parties is therefore an important part of this process. In all cases, the organisation must ensure that the organisation and processes are defined with scalability and prompt reaction in mind as this is required for due-diligence The Program Office has a central role in organising this set of processes.

The core of the process is articulated on two series of coordinated suites of processes: the handling of complaints, which are externally generated, and the handling of incidents, which are detected by operational monitoring or are the result of complaints after investigation and qualification.

- Handling complaints: the organisation must be ready to:
  - Receive, handle, and respond to complaints and information requests in a timely manner as required by internal policies or legal requirements
  - Use a case management system to support the handling of requests, track progress, and provide global statistics for use by both the operational management, the Program Office, and the Board.
  - Have a FAQ to anticipate the most frequent information requests
  - Create a classification for requests and complaints. Define standard procedures for the handling of requests in each of the classes
  - Have defined escalation procedures

- Handling incidents and breaches: the organisation must be ready to (in sequential order):
  - Log and track the incidents in a secure, time stamped and reliable way
  - Provide a prompt operational response to the incident (“stop the bleeding”). This can mobilize staff from all business and technical departments of the organization as well as external experts
  - Notify the stakeholders (affected parties) of the incident, providing remediation actions, as soon as those have been identified
  - Report the incident to authorities when required by law or by the criticality of the incident – including both regulators and law enforcement
  - Perform a root cause analysis
  - Repair the affected applications and restore the business processes in full
  - Build and distribute an account for the incident, which in particular attributes the failure corresponding to the incident

- Preparedness: in order to be ready to perform the above mission, the organisation must:
  - Define the relevant processes and procedures
  - Allocate the responsibility and deploy the necessary staff
  - Deploy the required tools (e.g. case management and incident tracking)
  - Have a contingency plan to deal with events of an exceptional magnitude
  - Place required external resources on retainer, to handle incidents of an exceptional magnitude or the provide forensics expertise
  - Get insurance against risks (based on a risk / cost analysis)
  - Define metrics for the various processes, track performance, and report to the responsible organisations
  - Test the system based on simulated incidents
  - Regularly update these various elements to ensure they are current.

3.8 Audit and Validate

It must be noted that audits focus, in all or in part, on continuous improvement, providing a proactive view rather than solely looking to place blame. Audit requirements are typically mandated by the domains for which the organisation is accountable. There is however a pattern that can be identified:

- Internal audits
  - Regularly perform internal audits of the system.
  - Focus on both compliance to internal and external criteria.
  - Investigate the effectiveness of the risk treatment plan as implemented. Ensure that the objectives are being met.
  - Trigger a review and adjustment process when critical deficiencies (blind spots) are discovered
  - Prepare for external audits. Ensure that recommendations from previous external audits have been properly considered and dealt with.
  - Collect the material and perform the analysis which will allow to prepare updated accounts, to include actual indicators of effectiveness rather than solely relying on the theoretical analysis used in the Analyse and Design phase (see section 3.5)

  The internal audits are performed by internal auditors, who work within the organisation and report to the audit committee.

- External audits
  - External audits are performed as dictated by the regulations of each domain of accountability.
  - External audits are also required in most compliance or attestation frameworks.
  - Customers may also place contractual provisions for realizing audits. The industry trend, in particular as it regards Cloud Computing, is to define certification and attestation to best-practice frameworks which can be effective substitutes for client-directed audits.

  The external audits are performed by external auditors, which either perform on the basis of an auditing contract or are hired by an external party (e.g. stakeholder or certification authority).

3.9 Demonstrating Accountability – The Account

Once the Accountable Organization has implemented the required accountability mechanisms, the next step in the accountability lifecycle is the provision of the *account*. As stated in [3], "*an account can simply be defined as 'a report or description of an event'.* Such an event can either be expected as "prescribed" by one of the organization's obligations or it can be an incident such as a breach or failure and thus unexpected. The previous definition can also be extended in order to cover *proactive* reports: the accountable organization can also generate some reports prior to making the service available: such a report may indicate the "quality" and "security" level of the services and may include certifications on practices or some data protection impact assessment. Another essential element of the proactive account is of course the policy describing the normative, regulatory and contractual obligations agreed between the Cloud Provider and its customer. By analysing the policy only, the Auditor, the Data Protection Authority or even the Data Subject may discover some issues or inconsistencies.

In addition to proactive reports, the Accountable Organization should also report while its services are operational. In this case, the Accountable Organization will either validate its operations or inform on an incident. As explained in [3], while describing such an event, be it expected (legitimate event) or unexpected (incident), " *the account should generally include the answers to what are traditionally referred to as the reporter's questions […] backed up with as much evidence as possible to validate the account".* These questions are listed in the following (we refer the reader to Section 4.3.2 for more information on evidence):

- *Who?* The account should provide information on all cloud actors involved in the actual event. This information will especially be very helpful for the Auditor or the Data Protection Authorities to identify the responsible or liable actor.
- *What?* The report should describe all actions taken with respect within this event or provide the details on the incident.
- *Where?* The answer to such a question is especially helpful while verifying the compliance with respect to data transfer policies.
- *When?* The account should mention the time (preferably a timestamp) and duration of the actual event.

While these four questions should definitely be answered in the case of both expected and unexpected events, the account on legitimate events may also include some more details on the process in order to demonstrate the compliance to the corresponding policy rule by answering the following two additional questions:

- *Why?* The answer to this question will simply be the obligation or policy rule the accountable organization is aiming at enforcing.
- *How?* The report should include as much details as possible on the means used to achieve the corresponding action. For example, to demonstrate that a Cloud Provider implements a security and privacy measures, it should provide all the details on the underlying functions such as the encryption algorithm, the size of the encryption key, etc.

On the other hand, although an account describing an incident cannot easily answer to the previous two questions, it should nevertheless provide some information on remediation and hence answer to the following question:

- *What Next?* In [3] authors note that an account is used in a *"prospective function"*; hence together with the description of the incident the account should ideally contain some additional information on future remedial actions and the adopted measures to prevent the recurrence of such an undesired event.

Although the description of the event or process is an essential element, the account should also carry the following attributes:

- The account recipient: This is the actor who receives the account. Depending on who the recipient is the level of details in the description of the event may change.
- Evidence (section 4.3.2)
- Timestamp and signature: The accountable organization is of course responsible for producing the account and therefore should sign the entire report including the date. Accounts on legitimate events may be periodic and could sometimes be used as evidence for prior events whenever an incident happens in the future. A timestamp in the report hence becomes mandatory.

3.10 The Role of Standards

There are many ways to classify standards. For example, in the A-5 work-package of this project, we notably distinguish *real* standards, from *technical specifications* and *best practices*. In the C-3 work-package of this project, we make another type distinction regarding the scope of standards, based on the four categories of standards defined CEN-CENELEC[2]. We will reuse this classification to structure the discussion of this section. If we restrict ourselves to the IT domain, these 4 categories can be expressed as follows:

1. **Fundamental standards** - *which concern terminology, conventions, signs and symbols, etc.*;

[2] http://www.cencenelec.eu/research/innovation/standardstypes/Pages/default.aspx

2. **Organization standards** - *which describe the functions and relationships of a company, as well as elements such as quality management and assurance, maintenance, value analysis, project or system management*, *etc.*
3. **Specification standards** - *which define characteristics of a product or a service, such as interfaces (APIs), data formats, communication protocols and other interoperability features, etc.*;
4. **Test methods and analysis standards** - *which measure characteristics of a system, describing processes and reference data for analysis*;

In the following paragraphs we examine the role and value of each category of standard as a driver for accountability for organizations.

3.10.1 Fundamental standards

Fundamental standards are like the foundation of a building; they are needed to create solid constructions. They play a role in setting the terminology and concepts that are used by organizations implementing IT systems and they influence other types of standards. From a strategic point of view, it is therefore important to include accountability as a crosscutting concept in fundamental cloud standards where relevant and possible. So far, most of the standardization initiatives in work-package A-5 has precisely been directed at putting accountability in core standards (see [15] for details).

3.10.2 Organizational standards

Organizational standards and Specifications standards form a complementary pair. In a simplified view, we can argue that organizational standards are useful to structure the internal processes of an organization to take accountability practices into account. By contrast, we can also argue that specification standards, by promoting interoperability, enable accountability across the supply chain with external entities. We will first discuss the role of organizational standards.

Organizational standards are not strictly necessary to enable accountability practices within an organization. In theory, this goal can be achieved by applying best practices that have been developed internally. Such practices could notably be inspired by the A4Cloud conceptual framework [3]. This approach has some important drawbacks however. First, it makes it complex for external entities to evaluate the quality of the accountability practices implemented by the organization. Second, it makes comparison between organizations largely impossible, since each organization will be using its own logic and criteria. Standardized approaches solve these two problems by structuring practices in a way that is recognized not only within an organization but also across the whole industry. In addition, such standards can be used as a foundation to build certification schemes, with independent third party auditors, with the benefit of recognition and enhanced trust. This could create a market for "accountability" certification, much like existing ISMS certification today.

There essentially are two competing approaches to embed accountability into organizational standards:

1. Take existing standards in security, governance and compliance, identify their gaps regarding accountability and extend them if needed to cover these gaps.
2. Build a new "accountability management standard", mirroring ISO 27001 for security for example.

Both approaches have advantages and drawbacks.

Taking an existing organizational standard and extending it to cover accountability practices allows organizations to re-use a framework they already know. This normally minimizes the cost of "adding accountability" to current practices, which in turn facilitates adoption of accountability practices. The Cloud Control Matrix[3] (CCM) is an example of an organizational cloud control framework that uses this attractive approach: all CCM control reference back to existing equivalent controls in other frameworks when they exist (in ISO/IEC 27001, PCI-DSS, ISACA COBIT, NIST, etc.). Using this approach for accountability means however that accountability is "added" to current practices and is not the backbone of the organizational practices. Building a real accountability organizational standard from scratch would

[3] https://cloudsecurityalliance.org/research/ccm/

allow describing governance, risk and compliance processes that would be structured around accountability. Building such a standard with industry consensus is however a huge task in itself.

3.10.3 Specifications standards

Specification standards allow accountability to be expressed and transferred along the supply chain, by promoting common metrics, common semantics, common data formats. This ultimately leads to automation of accountability interactions, in turn bringing cost reduction, which makes the value proposition of accountability more attractive. These points are extensively discussed in task C-3 [16].

3.10.4 Test methods and analysis standards

This last category can be exemplified through software testing standards such as [17], but is less relevant to our work on accountability, so we will not discuss it further.

4 Implementing Accountability across the Supply Chain

4.1 Challenges in Implementing Accountability across the Supply Chain

Cloud computing describes a model for enabling ubiquitous, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction [18]. Its key characteristics are on-demand self-service, broad network access, resource pooling, rapid elasticity, multi-tenancy (of users and/or applications) and measured service. Cloud computing can be provided via different service models, such as *Software-as-a-Service* (SaaS), *Platform-as-a-Service* (PaaS) and *Infrastructure-as-a-Service* (IaaS), as well as deployment models such as *private, hybrid* and *public* cloud [18].

A major benefit of cloud computing is that it enables the flexible composition of powerful applications by chaining together functions provided by different cloud services. For example, end-user-facing cloud applications may be composed from different service components packaged as Software-as-a-Service offerings, themselves utilizing cloud resources provided by different Infrastructure-as-a-Service providers. Furthermore, the use of standard interfaces and technologies means that cloud services[4] along the service provisioning chain may be substituted with others of similar specification without radically altering the way the application is composed.

An implication of this model however is that separate, independent entities assume control, ownership and responsibility for different parts of the service provision chain, the latter constituting separate domains of control. This is illustrated in Figure 4 below, which presents a typical cloud service supply chain. A cloud service provider is operating a datacentre to provide a public IaaS cloud offering. Numerous tenants utilize the cloud resources made available to provide applications to the general public in the SaaS model. Each tenant's virtual environment is isolated from all others' by means of the IaaS provider's virtualization and management infrastructure. Finally, customers access tenant applications over the public Internet to support various business functions.

Clearly, different parts of this supply chain belong to different control domains. The IaaS cloud operator has administrative control over the IaaS support infrastructure. This is indicated in the figure by the area marked by the red dotted line. Each tenant only controls the virtual environment made available to them, such as the green or blue areas in the diagram. Finally, an application user may be responsible for the handling of data processed by tenant applications as part of some business function.

Based on the fundamental accountability practices identified in the A4Cloud conceptual model of accountability and discussed earlier in the text, being accountable largely means for an organization to implement the controls appropriate for the service offered, demonstrate that obligations stemming from policies and regulations are met, and handle exceptions appropriately, remedying failures when applicable.

Even in the relatively simple example of a cloud service provisioning chain presented earlier, the challenges involved with achieving accountability end-to-end, across the entire chain, are evident. Since every part of the chain is separated by boundaries between domains, it is hard for any actor to establish whether the processes and operations executed beyond its own domain are according to the agreed rules and obligations. Thus, even if a number of actors have individually implemented accountability mechanisms inside their own domains (implementing the accountability governance approach described in the previous section) there are no means for overall accountability to be extended to cover the entire supply chain.

[4] This is especially common for services operating at the IaaS layer, as the functions supported by different vendors at this layer are generally uniform.

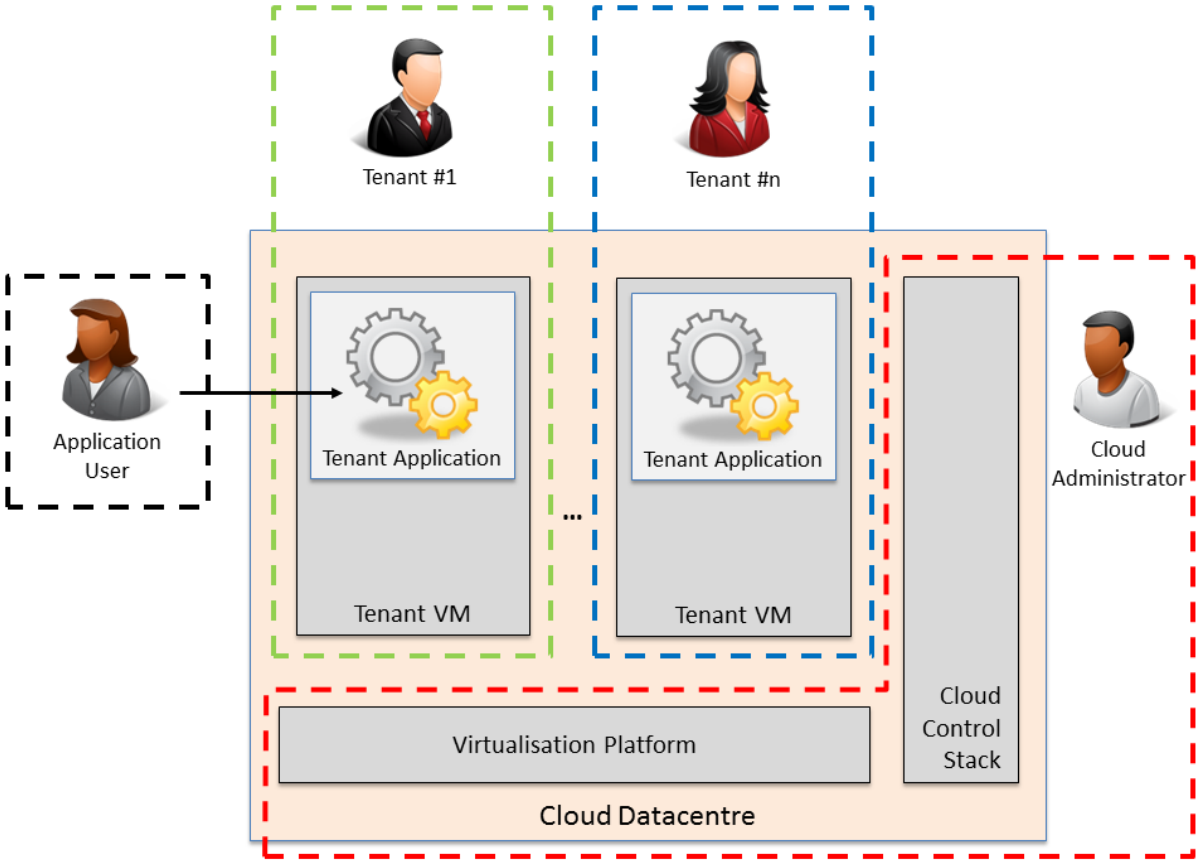

Figure 4: Separate domains of control in cloud service provisioning chains.

The A4Cloud Cloud Accountability Reference Architecture (RA) was developed to provide a method to tackle those challenges by implementing mechanisms to support accountability both within an organization and across cloud service provision chains. Before examining specific mechanisms, a number of key elements of the architecture need to be defined. First, the types of information objects that need to flow across the supply chain to support accountability are identified. Finally, the service-oriented approach for accountability in the cloud promoted by the RA is presented.

4.2 Flow of Accountability Information

In this section, we identify the types of accountability-related information that need to flow between different control domains and across the cloud supply chain to support end-to-end accountability.

Following the A4Cloud accountability model (presented in Chapter 2), we can classify all accountability-related interactions between actors[5] in a cloud supply chain into four fundamental types: Agreement, Reporting, Demonstration and Remediation.

- **Agreement** covers all interactions that lead to one actor taking responsibility for the handling of certain data provided by another party according to a certain policy. These interactions may include a negotiation phase. Agreement interactions require both (i) means to express data-handling policies, and (ii) means to describe implementation of policies.
- **Reporting** covers all interactions related to the reporting by an actor about current data handling practices (e.g. reporting incidents on consumer data) and policies.
- **Demonstration** covers all interactions that lead to one actor demonstrating the correct implementation of some data handling policies. This includes external verifications by auditors or cryptographic proofs of protocol executions for example. Demonstration is different from Reporting in that it implies some form of proof or provision of evidence.

[5] For the list of actors and roles identified in the RA, see Section 2.1.

- **Remediation** covers all interactions that lead one actor to seek and receive remediation for failures to follow data handling policies.

As part of each interaction a number of accountability objects are exchanged. During agreement, obligations, for example, need to be expressed into data-handling policies and passed for enforcement. Reporting involves the flow of information objects supporting transparency, such as logs, audit reports or incident notifications, to various recipients. Demonstration is largely about the provision of evidence to various parties in the supply chain. Finally remediation requires the flow of supporting information between the relevant parties to facilitate the appropriate remedy. We will now discuss these concepts in more detail.

Before a particular service is procured from the cloud, the prospective cloud customer needs to define its privacy and security requirements for the service. The requirements express the business (i.e. functional) needs of the cloud customer and may also contain elicited preferences of the cloud subjects it represents. In addition, the requirements will encompass any requirements stemming from law and regulation, such as limitations on how personal data are collected and handled. These requirements comprise the first type of accountability object, which is (implicitly or explicitly) made available to the selected cloud provider[6] as part of the service procurement process.

The cloud provider has requirements of its own, stemming from how its business is run and the constraints it has from law and regulation as well. These requirements inform the provider's service specification, which may be published in various forms ranging from documents outlining "terms & conditions" to machine-readable service description documents.

Typically, if the procurement process deems that the cloud customer's requirements are compatible to the cloud provider's service description, a contract is formed between the two parties describing the service agreement and responsibilities of each party to the other. In an accountability-based approach this step is particularly important because it is where the cloud provider's obligations to the customer are defined and agreed upon.

In general terms, obligations can be legal, contractual or normative [3]. Legal obligations cover accountability obligations imposed by law and regulation, such as those which arise from the Data Protection Directive and the proposed Data Protection Regulation (currently undergoing the EU legislative process)[7]. Contractual obligations are derived from contractual agreements. Finally normative obligations are standards with which legal and/or natural entities are expected to conform because they are moral agents. Obligations belonging to these three categories are combined to produce the full set of obligations agreed upon by the cloud customer and provider. This set of obligations comprises the second type of accountability object that must be generated.

The agreed obligations at this stage may be expressed in a variety of forms, with natural language being the most common. These obligations need to be translated into specific policies to be enforced by the cloud provider. This necessitates a procedure for the translation and conversion of obligations into machine-readable policies that can be automatically monitored and enforced so that all handling of data is performed according as prescribed by the policies. These (enforceable) policies comprise the third type of accountability object that must be generated.

During the operation of the cloud service, various elements of it will access, process or otherwise handle personal data at some capacity. While regular operations may be expected to handle the data as prescribed by the relevant policies, an accountability-based approach requires the explicit provision of

[6] To avoid unnecessarily complicating this example, we describe a process between two parties, a single cloud customer and a single cloud provider, which results in a straightforward agreement. Obviously, while searching the market for the most appropriate service offering, the cloud customer may engage with multiple providers in parallel, expressing its requirements to each of them and evaluating all responses before selecting one. Additionally, it may engage in negotiation of terms, which will imply a number of out-of-band iterations before agreement is reached. These actions do not affect the ultimate outcome of this process, which is the acceptance of agreed obligations by the provider.

[7] Section 8.1 lists the obligations stemming from the Data Protection Directive to which Cloud actors must adhere.

evidence to demonstrate compliance and meeting of obligations. Furthermore, information on the internal workings of the service, such as logs, may need to be provided for the purposes of auditing or to support transparency reports. Both items are thus important accountability objects that need to be generated and transmitted between different actors. In addition, as discussed in Section 4.3.2, the evidence collection process drives the development of the "account", which is the principal object for supporting accountability at various phases of the accountability governance lifecycle in its own right.

If an incident occurs or otherwise a violation of the agreed obligations is detected, the cloud provider must notify the affected parties, take steps to remedy the problem, and potentially offer redress. Thus, the construction of the notification comprises another accountability object, as does the provision of the account that the proper remedial steps have been taken.

Table 3 provides a summary of the various types of accountability objects discussed.

Accountability Object	Description
Obligation	Document-type object which refers to the legal, contractual and normative obligations, representing a human readable (natural language) form of the assigned responsibilities to the cloud computing and data protection roles.
Privacy and security requirements	Document-type object which refers to end-user preferences and organisational level policies, including security and privacy mechanisms that should be implemented.
Accountability policy	Machine-readable object which expresses the obligations and requirements for building a cloud service chain in machine-readable rules and policies.
Machine-generated logs	Machine- or human-readable objects, which are collected from various components of the cloud provider infrastructure (such as the network, hardware, the host operating system, hypervisor, virtual machines and cloud management systems, applications, etc.), detailing the actions that occurred during the execution of a service.
Evidence record	Structured information object, which aggregates information from logs with other metadata, any policy reference and the connection to previous records.
Audit report	Document like object, which aggregates evidence for a given actor per type and provides a collection of correlated records and their interconnected objects (i.e. logs, policies) to demonstrate compliance
Notification	An object meant to alert affected parties on the occurrence of an incident. It may contain relevant information on the incident, along with any potential corrective actions to be undertaken.

Table 3: Accountability objects.

These types of accountability information are exchanged during the development of the relevant accountability flows. We need to distinguish between an accountability information flow and a regular data flow created as part of normal cloud service operation. The accountability information flow is created to circulate the accountability objects described above, which are produced in response to a cloud service data flow. In principle, we can identify two types of accountability information flow:

- Accountability Information Flow that runs in parallel with a cloud service data flow (see Figure 5); in this type of accountability information flow, the data and the accountability information have the same direction from the data originator to the data recipient. Typically, this type has to do with the implementation of the preventive accountability mechanisms. As an example, the accountability policy is "attached" to the personal data as a sticky policy and follows the data flow as a set controls to guide the conditions, under which these personal data are managed.
- Accountability Information Flow that runs in response to a cloud service data flow (see Figure 6); in this type of accountability information flow, the actions applied on the data involved in a business transaction, which is governed by an accountability policy, can generate a set of reactions that are populated on the other direction of the data flow. As an example, an improper transfer of personal data to geographical areas not permitted in the policy should generate a violation report, which should be passed as a notification back to the actor shared these personal data.

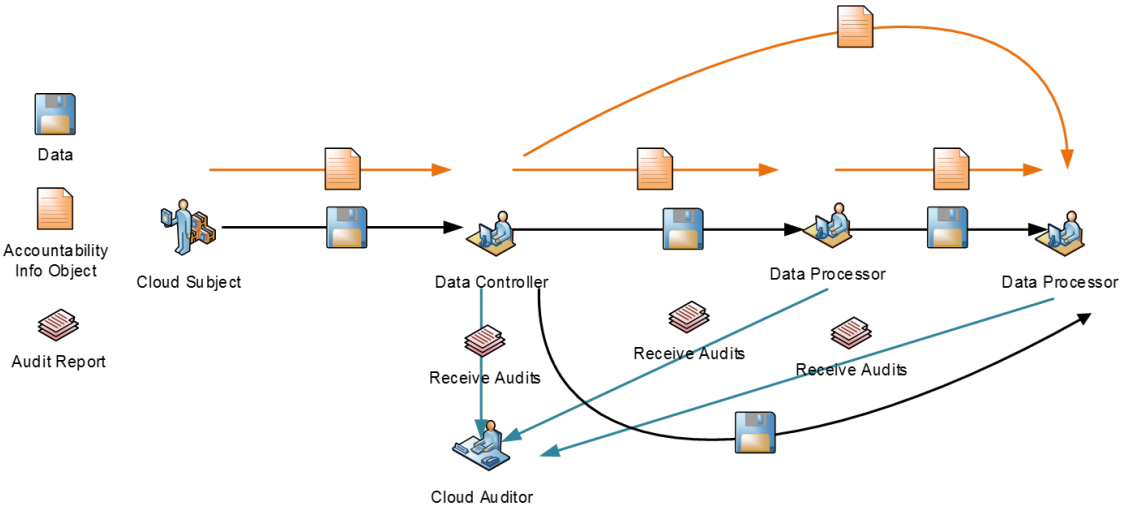

Figure 5: Flow of accountability information, in parallel with data direction.

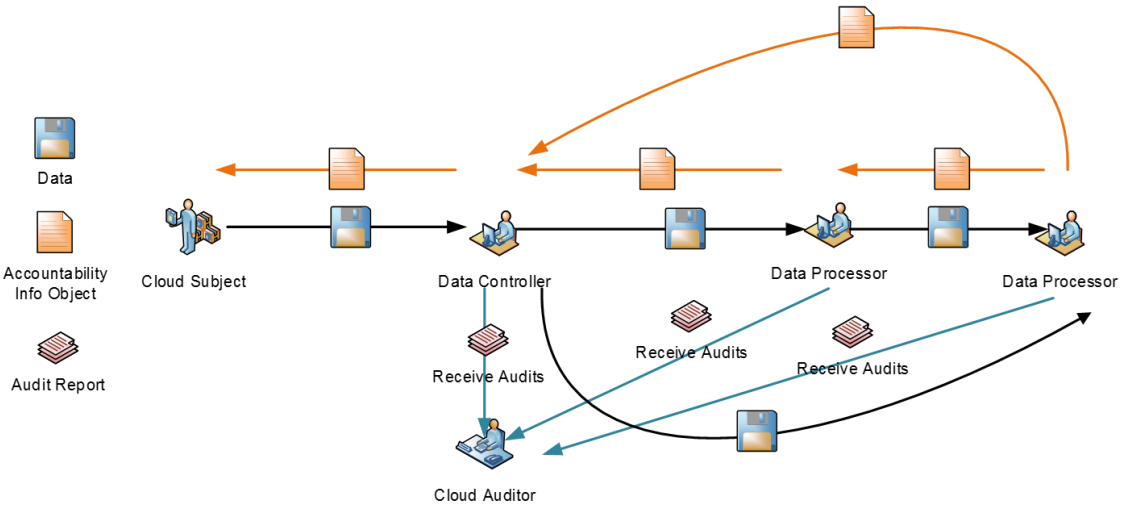

Figure 6: Flow of accountability information, in response to data direction.

4.3 Service-Oriented Approach for Accountability in the Cloud

In the introduction to this chapter it was established that cloud service provision chains are composed from heterogeneous elements which reside in separate control, trust and ownership domains. This makes the task of supporting accountability across the chain difficult, as accountability objects (described in the previous section) need to be generated and exchanged unaffected by the boundaries formed between different domains.

It becomes clear, therefore, that any mechanism supporting accountability cannot rely on centrally-controlled or centrally-organized functions, as these will be in conflict with architectural (i.e. by definition a SaaS provider does not have access to the same controls as an IaaS provider) or operational constraints (i.e. it is unrealistic to assume that an IaaS provider will allow full administrative access to a third party for security reasons, even if that party has a legitimate requirement for it) imposed by the existence of boundaries between domains.

Instead, we are proposing a service-oriented approach for supporting accountability in the cloud. Specifically, we have identified a set of high-level functions (services) which participants must implement to support accountability across a supply chain. These are:

- Policy definition and compliance
- Policy enforcement
- Collection & management of logs and evidence

- Creation & management of the account
- Notification
- Remediation
- Data subject enablement

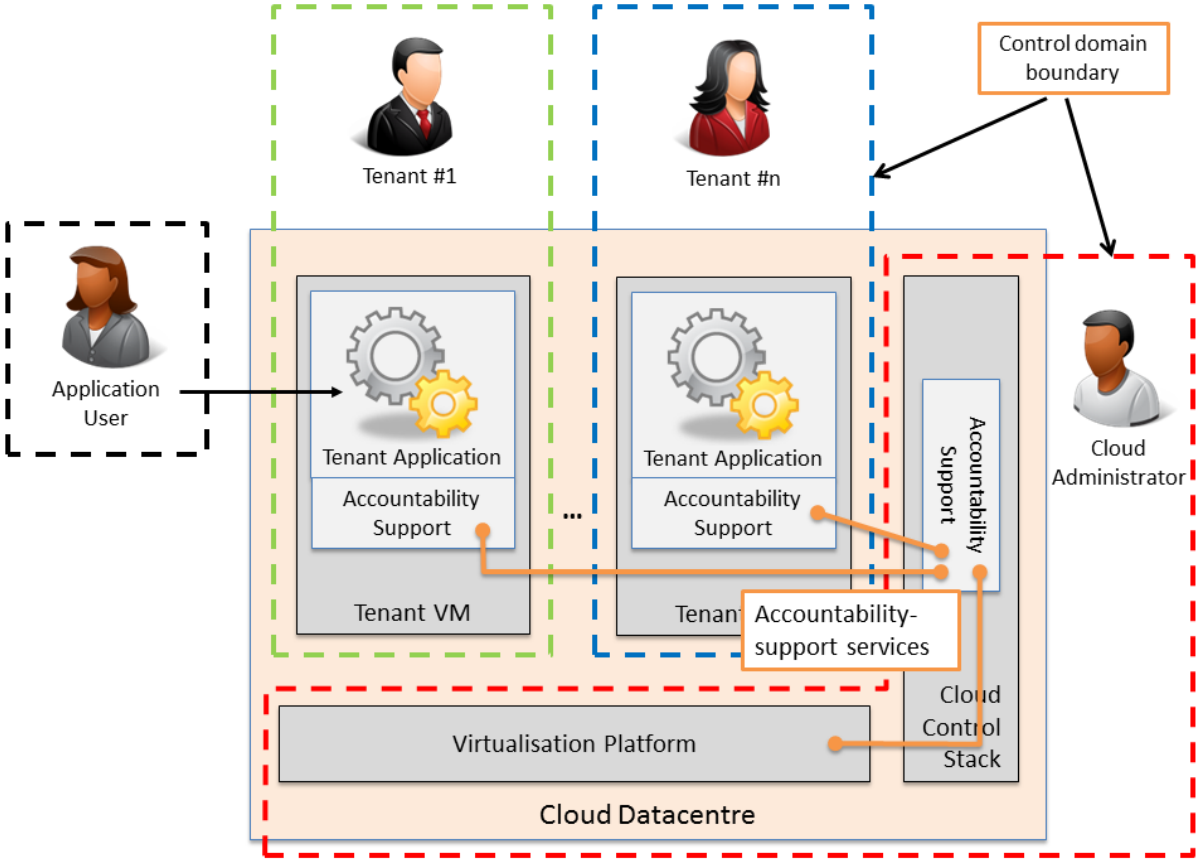

Figure 7: Supporting accountability via a service-oriented approach.

The A4Cloud tools are specific instantiations/implementations of these services or elements of them, leading to a clear correspondence between accountability services and the A4Cloud toolkit architectural areas discussed in the next chapter. The rest of this chapter will focus on analysing further these classes of accountability support services, grouping together those that are complementary.

4.3.1 Policy Definition, Compliance and Enforcement

In general terms, policies in IT systems specify sets of rules related to a particular purpose, such as defining the security credentials one must possess to access a particular data object and the actions to be taken under various conditions. In the scope of the A4Cloud project the obligations an organization has in regards to how it must handle personal data are also expressed through policies.

Organizations that are subject to obligations need to not only meet their obligations, but also to ensure that their business partners and sub-contractors do not invalidate them. In particular, an accountable organization needs to make sure that their obligations to protect personal data are adhered to all across the service provisioning chain.

An important aspect of governance in an accountable organization is to define and deploy policies for their data processing practices and to make sure that they are followed by all the involved service providers. The policies should ideally travel with the data, and they should be used as input to monitoring of data processing practices, to generate evidence that policies are fulfilled, to correct policy violations that may occur and in general to demonstrate policy compliance.

Therefore, an accountability-based approach requires services via which to express accountability obligations using a common policy specification language (or standard interoperable policy specification languages) and to distribute them throughout the cloud supply chain. An additional component of policy definition support is the checking of compliance between the original obligation and the actual set or rules and actions defined in the policy.

A separate but closely related set of services is concerned with the enforcement of policies. In order for obligations to be met, the policies which express them once these are defined and enacted need to be enforced using various mechanisms and tools. In this section we consider both policy definition and enforcement service support.

The A4Cloud project has developed a cloud accountability policy representation framework [19] (see Figure 8 from the same source) based on the following design requirements:

- Access and Usage Control rules - express which rights should be granted or revoked regarding the use and the distribution of data in cloud infrastructures, and support the definition of roles as specified in the Data Protection Directive, e.g. data controller and data processor.
- Capturing privacy preferences and consent - to express user preferences about the usage of their personal data, to whom data can be released, and under which conditions.
- Data Retention Periods - to express time constraints about personal data collection.
- Controlling Data Location and Transfer - clear whereabouts of location depending on the type of data stored and on the industry sector processing the data (subject to specific regulations) must be provided. Accountability policies for cloud services need to be able to express rules about data localization, such that accountable services can signal where the datacentres hosting them are located. Here we consider strong policy binding mechanisms to attach policies to data.
- Auditability - Policies must describe the clauses in a way that actions taken upon enforcing the policy can be audited in order to ensure that the policy was adhered to. The accountability policy language must specify which events have to be audited and what information related to the audited event have to be considered.
- Reporting and notifications - to allow cloud providers to notify end-users and cloud customers in case of policy violation or incidents for instance.
- Redress - express recommendations for redress in the policy in order to set right what was wrong and what made a failure occur.

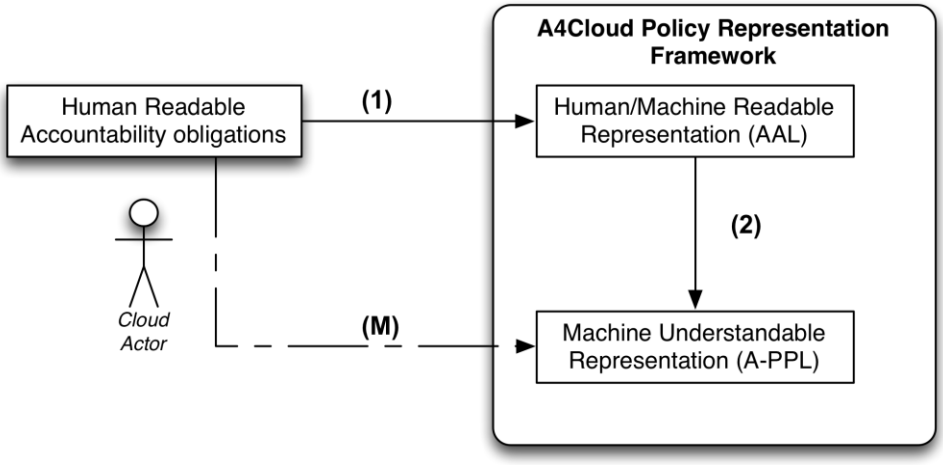

Figure 8 - Overview on the accountability policy representation framework.

While there are languages for privacy and theoretical models for accountability, we proposed AAL, a domain specific language (DSL), to express abstract obligations in a language close to sentences in laws, data directives and contracts. This language is equipped with a formal logical background and is the first stone towards obligations enforcement trough accountable design and verification [20]. We also

define a concrete policy enforcement language, called A-PPL, as an extension of the PPL language. The proposed framework offers the means for a translation from abstract obligations expressed in AAL to concrete policies in A-PPL.

In the A4Cloud policy framework, the cloud customer must define the high-level data-handling requirements and obligations in AAL, a representation which is both human- and machine-readable. These policies can be communicated and agreed with potential cloud providers. Since AAL has an associated formal semantics and there are tools to check its properties using a model checker which is embedded in the AccLab tool[8] it is possible for cloud customers to use the tool to check cloud provider policies for conformance with their own data governance practices.

If the cloud provider uses subcontractors, then it must check the conformance of the cloud customer policy with the practices of the actors further down the chain. In Figure 9 we illustrate a cloud service chain. The SaaS provider must propagate *Policy 1* from the Cloud Customer to an adapted form to manage the resources it uses from its subcontractor. AAL and AccLab have the necessary features to help in the definition and verification of these dependencies. Since AAL is abstract, it acts as a pivot model between different accountability obligation representation models and such promotes the interoperability of heterogeneous systems.

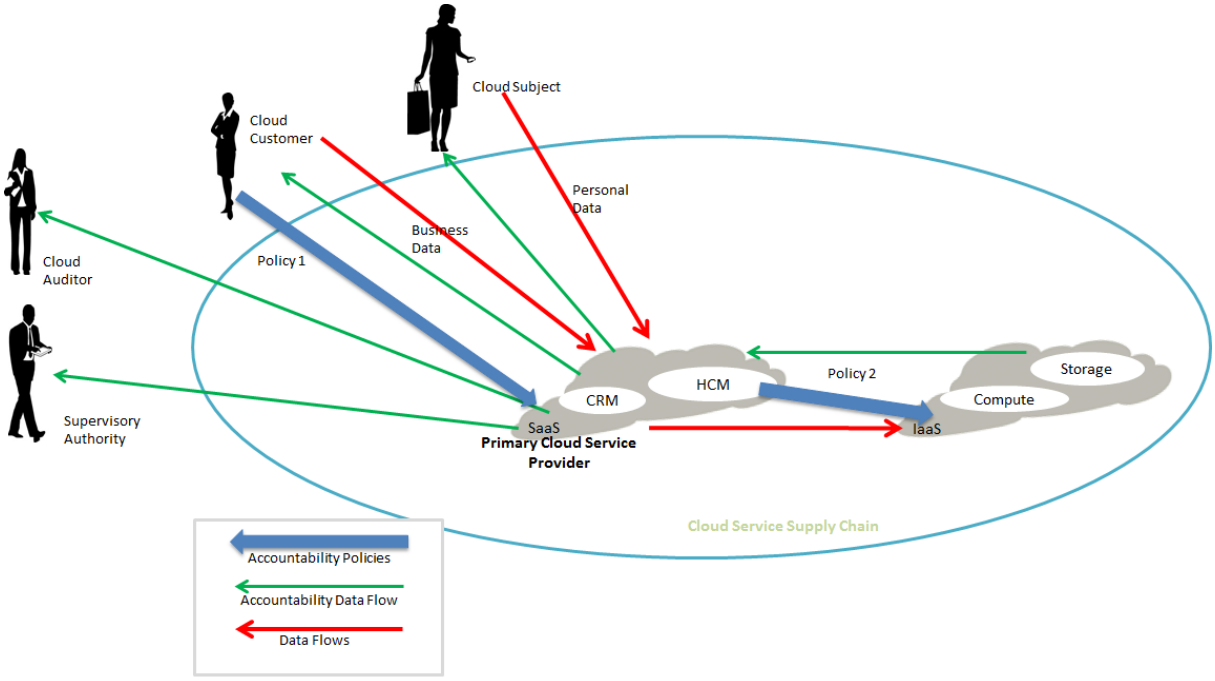

Figure 9 - Accountability Policy Distribution and Accountability and Data Flows

At a later phase, the accountability obligations expressed in AAL are (semi-)automatically translated into a machine understandable policy (Step 2 in Figure 8). The main target of the mapping step is A-PPL, for expressing usage control and data handling obligations [21]. However the framework is flexible enough to accommodate further target languages and formats. This flexibility is important to set up the different mechanisms making part of the A4Cloud toolset, involving preventive, detective, and corrective tools.

Accountability policy enforcement can be automated with the help of the A-PPL-Engine to control resources in any service delivery model, as soon as it is correctly configured and integrated to the CSP technological landscape. This can be achieved through an auditing process to verify that the CSP has integrated the tool (or more generally the A4Cloud toolkit) as a service correctly as recommended by the methodology exposed in the present deliverable.

[8] http://www.emn.fr/z-info/acclab/

4.3.2 Evidence and the Account

The provision of evidence is an essential element of an accountable system, enabling demonstration, verification and the construction of the account, while motivating and guiding a range of other functions and procedures important for accountability. The A4Cloud project defines accountability evidence as "*a collection of data, metadata, and routine information and formal operations performed on data and metadata, which provide attributable and verifiable account of the fulfilment of relevant obligations with respect to the service and that can be used to convince a third party of the veracious (or not) functioning of an observable system*" [22].

The project identifies the following five major types of evidence relevant for accountability:

- **Data processing practices**: evidence on operational practices such as replication, storage, deletion, copy, access, optimization, consent, security, segregation, proofs of retrievability, etc.
- **Data collection practices**: evidence on data collection practices such as policies compliance, privacy issues, security breaches, etc.
- **notification**: evidence that notifications were sent to the interested stakeholders in case of privacy issues (unauthorized access, etc.), policy violations, security breaches (data leakage, data lost, corrupted or tampered, etc.) and services or policy modifications, as well as service practices and users rights;
- **Remediation**: evidence on remediation to their customers in case of security breaches, privacy issues and policy violations.
- **Organisational practices**: evidence on requirements related to employee's training, system certifications, privacy policies, etc.

The provision of evidence is a complex process that entails identification, association and collection of information that will constitute evidence from various sources; processing and use of cryptographic techniques to ensure integrity; secure storage; and presentation for the various different stakeholders, auditors and regulators. Specifically, the process must be designed in such a way as to:

- Give account of events and occurrences within the services of cloud providers in a non-invasive manner to the customers' privacy and without exposure of sensitive material from the inside of organisations;
- Ensure information is collected securely in a tamper-evident process, which however may allow verification checks from different services, tools and trusted third parties (e.g. auditors);
- Avoid excessive data collection, minimising privacy issues and ensuring scalability of evidence storage with time, even with growing user bases and multi-tenancy scenarios.

To address this challenge the A4Cloud project has designed a Framework of Evidence (FoE) consisting of a set of mechanisms for extracting evidence in typical scenarios encountered in cloud services and managing their full lifecycle. The FoE considers three main types of sources:

- internal data as logs, metadata SIEM outputs, etc., collected from monitored operations and services;
- data from end-users or from a cloud service to another, or internal data as customers' or employees' lists, training certificates, seals, etc.;
- documentation of the service's practices, contracts and organisational policies that relate to accountability and for which implied obligations can be expressed (manually or by other A4Cloud tools like AccLab or COAT – discussed later in the document) in machine-readable policies.

Using the FoE, the collected and processed evidence are assembled and stored as *records*, and can be utilized for account provision, attribution of responsibilities and policy violation reports. Records contain elements supporting a claim such as the actions that the evidence records, a timestamp, the identity of the authenticated agent that performed the recorded action and a reference to the policy that the action may or may not comply with. Cryptographic proofs are also gathered in the records as evidence that compliance checks have been made upon request at a particular point of time.

Therefore, to support accountability across the supply chain, services providing the following high-level functions need to be provided for evidence provision:

- Support for the extraction of evidence targets (i.e. what to monitor, when, etc.) from policies, ensuring compliance and full coverage (i.e. no obligations described in policies are left unmonitored);
- Support for collection of logs from different sources and parts of the infrastructure;
- Support for full lifecycle of evident management (i.e. mechanisms to extract, assemble, secure, store evidence, etc.), ensuring privacy, integrity, verifiability;
- Support for enforcing access rights on evidence;
- Support for context-specific presentation of evidence to different parties possessing different access-rights and privileges;

Evidence are also supporting elements for the provision of the account. The C-2 conceptual framework [3] defines three types of accounts that can be provided. We outline here some examples of evidence that can be provided to support these three different types of accounts:

- *Proactive account*: This kind of account may report the quality and security levels of the services provided by the cloud. Evidence in this case may consist of certifications of the compliance of the offered data processing and storage services with regulations and contracts. The certificates should designate the party that provides the service, the issuer of the certificate, the validity period of the certificate, the level of security guaranteed by the certified service, etc.
  Additionally, proactive accounts can be supported by consistency checking reports that act as evidence of contractual obligations agreed between the Cloud Provider and its customer.
- *Account on legitimate event*: Evidence should be collected to provide an account of a legitimate event to demonstrate that the Cloud Provider complies with regulations and contracts. For example, Proofs of Retrievability, such as the one presented in [22], are cryptographic proofs that verify whether or not a data storage service actually stores the outsourced data. Collected on a periodic basis, these proofs of retrievability enable an auditor to check the correct storage of the data, meaning that the service stores the data as expected by regulation and contracts. Therefore, the proofs of retrievability can support this kind of account.
- A*ccount on incident*: In case of the occurrence of an incident, evidence should demonstrate that the fault actually occurred, identify the actors and the environmental settings that lead to the incident, assign time and date to the event, and have a reference to the particular policy rule(s) that the reported event infringe. Logs can be an example of such inculpatory evidence to provide an account in case of an incident and to attribute the responsibility of that incident to a particular actor in the cloud. Indeed, logs report the actions performed by a particular entity and record the identity of that entity and the timestamp corresponding to the reported event.

The RA does not prescribe a specific method for providing an account, recognizing that in many cases the form and contents of the account as well as the context of its provision are specific to the particular circumstances of the actors involved. For example, a particular account may consist of highly structured and annotated information allowing it to be transmitted electronically in an automated fashion, or may be an e-mail containing textual information that is not organized in a specific way.

However, given that the construction and provision of the account requires the provision of evidence, the RA notes that in order to support accountability across the supply chain via the construction and provision of the account, actors need to implement the services supporting evidence provision outlined above, as well as implement services supporting context-specific presentation of the account to different parties possessing different access-rights and privileges, in cases where the account can be processed automatically by computers.

4.3.3 Notification & Remediation

Notification is an essential element of accountability. A strong accountability-based approach requires cloud providers to notify all affected parties of the occurrence of an incident or discovered policy violation within a reasonable timeframe. Notifications may be provided through common means such as e-mail or letters to the relevant parties, or dedicated communication channels designated for the purpose (usually between cloud providers).

The RA does not prescribe particular methods for notification recognizing the fact that the circumstances around each incident rarely are the same and flexibility should be allowed in which mechanisms to

mobilize during response. However, the following functions should be implemented for a strong accountability-based approach:

- Obligations with regards to providing notification within a predefined, reasonable timeframe should be reflected in the policies enforced. As such, any policy-support services implemented (discussed in section 4.3.1) should ensure that this element is explicitly supported.
- An automated approach to transmission of notifications can reduce the burden of operating this part of the accountability lifecycle. As such, organizations may opt to implement services supporting the exchange of machine- and human-readable notifications based on a predefined protocol that has provisions for the inclusion of information about the incident (including evidence of which subset of a subject's personal data were involved in the incident) and, where possible, links to an automatic remediation management system, discussed next.

Like notification, remediation is also an essential element of accountability, referenced directly by the fourth and final accountability practice in the accountability model presented in Chapter 2. Again, the specifics of a particular remedial action in response to a specific violation depend on the circumstances of the violation itself, and many may be enacted ad-hoc. As such, the RA does not propose a particular mechanism for remediation. We do, however, note that a framework for the systematic addressing of violations and provision of remedies depends on the proper implementation of the accountability-support services described in the previous sections. Specifically, service functions should be in place to facilitate:

- the ability to detail the origin of policy violations in order to provide appropriate responses. Customers need to know whether the policy violation occurred as the result of an attack, a deliberate action by the provider, an unintended alteration or any other mean, in order to make an educated decision about the efficacy of the proposed remediation or request additional redress.
- the ability to suggest response actions to ease the process for customers responding to the event. Customers could get assistance from the service provider in performing any necessary step on their part to handle the event. This would include any remediation action deemed appropriate.

4.3.4 Data Subject Enablement

The final necessary element for end-to-end support of accountability across cloud supply chains is the provision of services aimed at enabling data subjects to consent, control, review and correct their personal data held in the cloud.

Specifically, the Data Subject should be provided with facilities to:

- provide consent on the use of their data;
- request from a data controller access to their data stored in the cloud for review;
- request from a data controller to correct or delete their data stored in the cloud;
- view detailed information about how the data has been shared and used by the data controller;
- receive notifications of incidents affecting them;
- receive assistance in requesting remediation and redress.

To support these functions, actors along the chain (excluding data subjects) should therefore implement services that implement the following functions:

- track and produce evidence of all data uses;
- provide means for data subjects to view their personal data held, along with meaningful metadata (e.g. time of data disclosure, etc.);
- provide means for data subjects to amend or request deletion of their personal data held;
- since almost all data subject controls will only interact with the data controller and not the actual data processors along the supply chain, services should be in place to facilitate the passing of data subject requests in an appropriate form to the relevant processors. Clearly, to support this capability the functionality provided by the accountability-support services described in previous sections (such as services to exchange enforceable policies and provide evidence) will be required.

5 The A4Cloud Toolset

The tools comprising the A4Cloud toolset have been designed to support accountability for data governance in the cloud by specifying certain functions identified during the development of the A4Cloud accountability framework. The tool design process was motivated by the goal to provide stakeholders with tools supporting those elements of accountability for which little or no support was found to exist, while complementing existing privacy and security mechanisms. Each A4Cloud tool addresses different elements of accountability, and may operate over different timescales and interact with data at different points of its lifecycle compared to others.

The A4Cloud toolset is composed of the following eleven (11) tools and a plug-in:

- The Data Protection Impact Assessment Tool (DPIAT): This tool is used by Small-Medium Enterprises (SMEs) to identify the risks in a given configuration and environment of carrying out a certain business transaction, such as buying a new cloud service.
- The Cloud Offerings Advisory Tool (COAT): This tool is designed to assist potential cloud customers (SME organizations and individuals) in assessing and selecting cloud offerings, with respect to certain security and privacy requirements.
- The Accountability Lab (AccLab): This tool translates human readable accountability obligations expressed in the Abstract Accountability Language (AAL) into our lower level machine-readable accountability policy language called Accountable Primelife Policy Language (A-PPL).
- The Accountable Primelife Policy Engine (A-PPL Engine): This tool enforces data handling policies and actions (e.g. logging), described in A-PPL.
- The Audit Agent System (AAS): This tool enables the automated audit of multi-tenant and multi-layer cloud applications and cloud infrastructures for compliance with custom-defined policies, using software agents.
- The Data Transfer Monitoring Tool (DTMT): This tool automates the collection of evidence describing how data transfers within a cloud infrastructure comply with data handling policies.
- Data Track (DT): This tool is used by data subjects to get a user-friendly visualization of all personal data they have disclosed to cloud service, with the additional capability to rectify data if necessary.
- The Transparency Log (TL): This cryptographic tool provides a secure and privacy-preserving unidirectional asynchronous communication channel, typically between a cloud subject and a cloud provider. Messages can be stored on untrusted system and can be still be securely retrieved asynchronously by recipients.
- The Remediation and Redress Tool (RRT): This tool assists cloud customers (individuals or SMEs) in responding to real or perceived data handling incidents.
- The Incident Response Tool (IRT): This tool is the entry point for handling anomalies and violations in cloud services, such as privacy violations or security breaches. The tool receives incident notifications and takes the initial steps to respond to these incidents, by sending alerts to the user and gathering comprehensive information related to the incident.
- The Assertion Tool (AT): this tool ensures the validation of the A4Cloud tools through a test case-based validation methodology, during the development and deployment phases.
- The Plug-in for Assessment of Policy Violation (PAPV): This is a plug-in component to the Data Track tool that provides an assessment on the criticality of previously detected policy violations. By using it, data subjects can check which policy violations are the most relevant ones to their data disclosed in the cloud.

It must be noted that at the time of writing this document, the design specifications of the RRT and IRT were in a preliminary phase. In the future, merging of the two tools is under investigation, but, for this document, the analysis is based on the assumption that they constitute two different tools.

At a high level, accountability functions can be separated into *preventive*, *detective* and *corrective.* Preventive functions focus on mitigating the occurrence of an unauthorized action. In the RA, preventive functions include assessing risk, identifying and expressing appropriate policies to mitigate it, and enforcing the latter via mechanisms and procedures put in place. Detective functions are used to identify the occurrence of an incident or risk that goes against the policies and procedures in place. In the RA, these centre on monitoring and identifying policy violations via detection and traceability measures such as audit, tracking, reporting, and monitoring. Finally, corrective functions are those that are used to fix

an undesired result that has already occurred. In the RA, these focus on managing incidents, providing notifications and facilitating redress.

Using this categorization, we can separate the A4Cloud tools into five functional areas:

- Contract and Risk Management: the A4Cloud tools in this area aim to address the need for supporting the management of risks and the selection of suitable cloud service contracts in the context of accountability for data governance in the cloud. All tools in this category implement preventive accountability functions.
- Policy Definition and Enforcement: the A4Cloud tools in this area aim to address the functionalities needed for defining and enforcing accountability policies, as well as their maintenance within the lifecycle of a cloud service provision chain. The tools in this category implement preventive accountability functions.
- Evidence and Validation: the A4Cloud tools in this area deal with the collection and provision of evidence and the validation of the proper execution of the accountability tools in a specific setting. The tools in this category implement both preventive and detective accountability functions.
- Data Subject Controls: the A4Cloud tools in this area target the needs of data subjects by providing controls for the proper management and protection of their personal data in a cloud service ecosystem. The tools in this category implement detective accountability functions.
- Incident Response and Remediation: the A4Cloud tools in this area provide corrective accountability functions facilitating remediation and redress.

Figure 10 illustrates the A4Cloud tools according to this classification. The rest of this chapter presents a detailed description of each A4Cloud tool, examining these functional areas in succession. The description of the tools is structured, so that, for each tool, the following information is provided:

- An overview of the tool, describing its main functions and the envisaged target stakeholders;
- The high level architecture of the tool, identifying its major functional components;
- The description of both the user and machine/application programming interfaces (UI and API) provided, indicating the input and output for each interface and the expected consumers, as well as the main required machine interfaces from other A4Cloud or external tools.

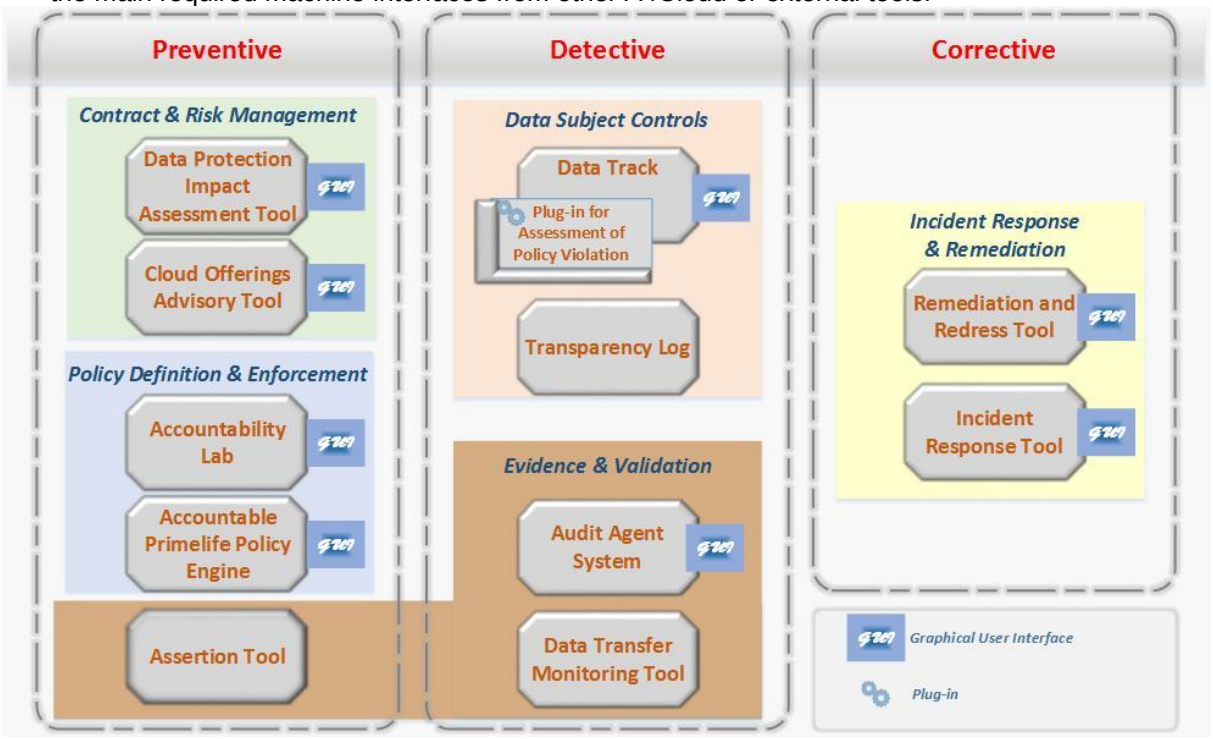

Figure 10: High level view of the A4Cloud Toolset.

5.1 Contract and Risk Management

The contract & risk management area of the A4Cloud toolkit architecture contains tools which address the need for support in managing risk and cloud service contract selection in the context of accountability

for data in the cloud. As a result, tools in this area serve a *preventive* role. Prevention is facilitated via two separate but complementary mechanisms, namely:

i. Assessment of the risks associated with various facets of the cloud service consumption process, involving personal and/or confidential data and elicitation of actionable information and guidance on how to mitigate them;
ii. Evaluation of cloud offerings and contract terms with the goal of enabling a more educated decision making on which service to select.

For the instantiation of this part of RA, these two mechanisms are being developed as distinct A4Cloud software tools: the Data Protection Impact Assessment Tool and the Cloud Offerings Advisory Tool.

5.1.1 Data Protection Impact Assessment Tool

The Data Protection Impact Assessment Tool (DPIAT) is used to identify the risks involved with carrying out a certain business transaction in a given configuration and environment, such as buying a new cloud service. The tool is used by Small-Medium Enterprises (SMEs) to assess the classification of the data involved in this business transaction (whether they are personal or sensitive data) and how they can be secured and protected in the cloud. Furthermore, DPIAT reports on risks with respect to data breaches and the privacy of the cloud service users. It, also, provides insight in the potential threats, associated with the detected risks.

The output of the DPIAT is a report that includes a risk profile document including advice on whether to proceed or not with the specific business transaction (in the context of the given configuration and the given details of the project), and the suggested mitigations in cases of risk exposure. It, also, logs the offered advice and the user's decision for accountability purposes. Further to it, the tool educates the user on risks and threats to ensure the ethical aspect of accountability.

DPIAT is used by SMEs to generate an impact assessment report, detailing the risks and threats associated with the specific setting defined for a given business transaction. In that respect, it integrates information coming from different sources, but the main source of information comes from the target end users (the SMEs), who interact with the tool through a questionnaire form. In order for the tool to work properly, some other end users are necessary to support the envisaged internal processes of the tool. As such, a domain expert and an approver can specify the workflow and the business processes governing the transaction under investigation, while the approver can verify the results of the impact assessment report.

Figure 11 shows the high level architecture of DPIAT.

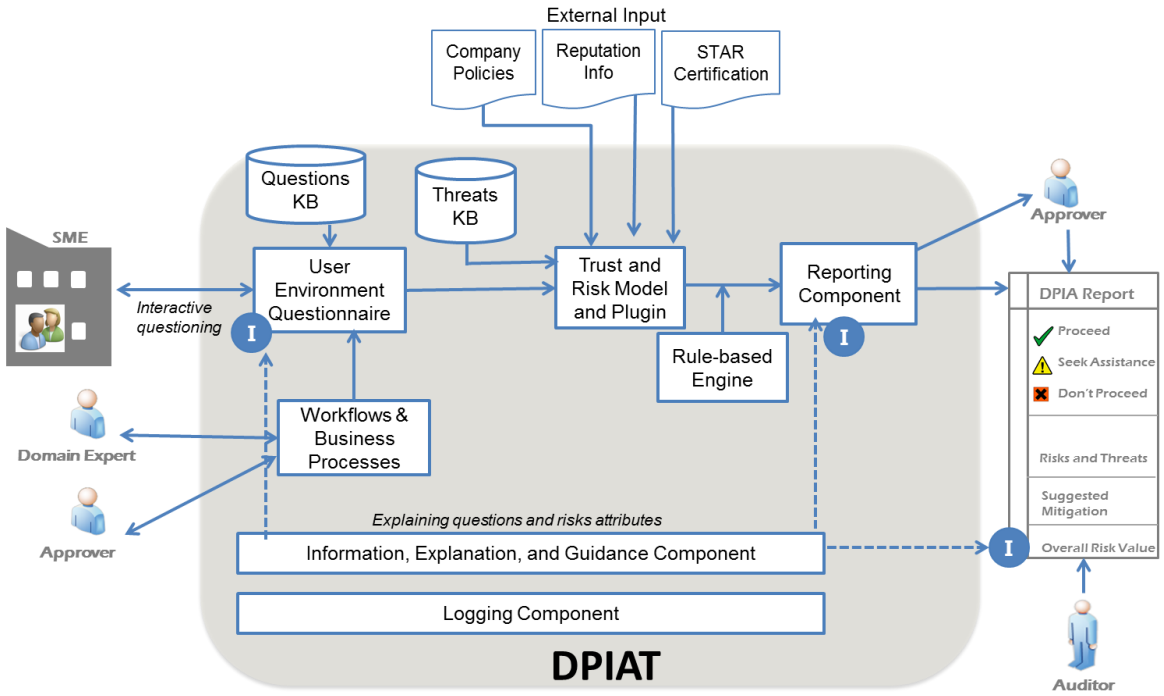

Figure 11: The high level architecture of the Data Protection Impact Assessment Tool

As shown in Figure 11 DPIAT is decomposed to the following main components, implementing respective process level mechanisms:

- The User Environment Questionnaire, which offers interaction with the users to collect information about the project (s)he that will be involved in the intended transaction (or the type of data that are going to be used in the transaction).
- The Trust and Risk Model and Plugin, which assesses the risks associated with the information collected from the User Environment Questionnaire.
- The Rule-based Engine, which sets up and processes the rules to the path taken when the user answers with specific answers.
- The Information, Explanation and Guidance Component, which provides support on the meaning of each question and the implications of the responses.
- The Logging Component, which logs the values of the answers to the questionnaire and the final report given to him.

The Reporting Component, which is responsible for presenting the user with a complete assessment report in a comprehensible and user-friendly format.
These components are fed with user related information (such as the data location, the roles involved in the transaction, contact details of those responsible for defining the purpose of use for the involved data), contextual information on the environment setting, the respective trust and risk modelling, external certification systems (with emphasis for our case the CSA STAR Certification program, which is to manually be adopted), reputation information with respect to the agents involved in the trust modelling, the organisational policies for the protection of the classified data and the knowledge base (KB) of the threats and their associated mitigation control actions.

Through this architecture, the DPIAT will be able to assess a set of use cases, including the support to decision making on competing cloud service offerings, the action on moving data into the cloud (or processes) and the determination on what services in the cloud are the best ones to choose.

DPIAT offers a Web User Interface, which enables interaction with the SMEs. During this interaction, SME representatives can feed the questionnaire with certain input, while the tool gives back a report with three colours (green colour means “Okay to move forward”, yellow colour means “Possible to move forward with issue resolution” and red colour means “Do not move forward - contact Approver”), the relevant threats and risks, and the suggested mitigations/actions and tracking.

DPIAT offers a single interface, named *Idpiat*, which has the following two methods, as shown in Table 4.

Table 4: Interfaces and respective API methods provided by the Data Protection Impact Assessment Tool

Name of the API/method	*Purpose of use*	*Consumed by*	*Data format*	*API format*
Idpiat / Retrieve Questionnaire	Returns a single instance of a questionnaire	The UI of DPIAT (target all envisaged users of the tool)	JSON	RESTful
Idpiat / List Questionnaires	Returns a list of all available questionnaires	The UI of DPIAT (target all envisaged users of the tool)	JSON	RESTful

Although these interfaces are currently used internally by the UI of DPIAT, it is examined whether they can be public so that they will be consumed by other tools as well in the future, through a web service messaging and transport layer (such as a RESTful service). A detailed analysis of the offered interfaces will be provided in a later stage.

DPIAT consumes the APIs provided by other tools and environments, as shown in Table 5.

Table 5: Interfaces and methods needed by the Data Protection Impact Assessment Tool

Name of the API/method	*Purpose of use*	*Consumed by*	*Data format*	*API format*
Countries List	Retrieve a list of countries that are available to Service Providers when adding their services. Used as options for the end user to choose from where to select their location	Cloud Providers	JSON	RESTful
Service Types List	Retrieve a list of service types that are available to Service Providers when adding their services. Allows the end user to filter offers by service type when selecting their questionnaire	Cloud Providers	JSON	RESTful
Questionnaire List	Retrieve a list of Questionnaires that can be chosen by the user to complete	Cloud Providers	JSON	RESTful

5.1.2 Cloud Offerings Advisory Tool

The purpose of the Cloud Offerings Advisory Tool (COAT) is to assist potential cloud customers (SME organisations and individuals) in assessing and selecting cloud offerings with respect to security and privacy requirements by providing information and guidance on:

- How to understand and assess what a cloud service provider is offering from a privacy and security perspective;
- How to compare offerings (from a data protection compliance and provider accountability point of view)
- The meaning of the comparison attributes.

The tool generates a guided comparison of the cloud service offerings, along with an explanation of the (selected) security and privacy attributes. The tool, also, logs the offered advice and the user's decision for accountability purposes.

COAT is primarily used by data subjects, both SMEs and individuals, who wish to become cloud customers. The tool assists such stakeholders in assessing the offerings of a cloud service provider offering, from a privacy and security perspective and compare offerings from various providers, from a data protection compliance and provider accountability point of view. It, also, provides guidance on the meaning of the comparison attributes and the education of users on security aspects, while it implements mechanisms, so that the offered advice and the user's decision are logged.

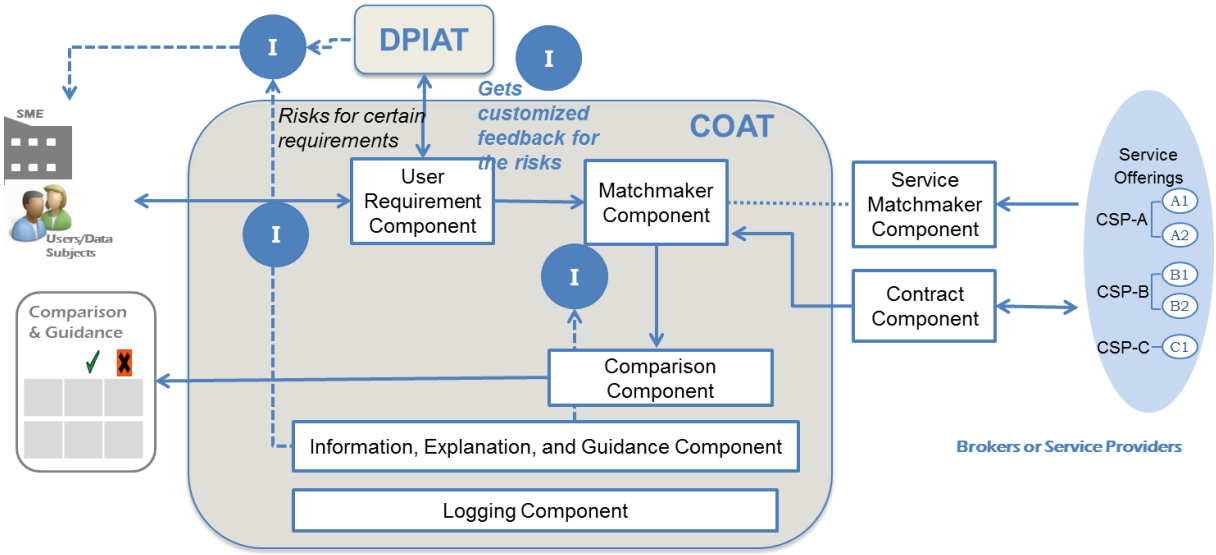

Figure 12: The high level architecture of the Cloud Offerings Advisory Tool

Figure 12 shows the high level architecture of COAT. As shown there, COAT is decomposed to the following main components, implementing respective process level mechanisms:

- The User Requirements Component, which gathers all the explicit (such as user requirements, based on criteria) and implicit (such as user location and contextual information) input for this tool.
- The Logging component, which logs the values of the user selections and the final report with the cloud offerings given to him/her.
- The Information, Explanation and Guidance Component, which supports the guidance of the users on the explanation to each term or criterion used in the comparison process.
- The Matchmaker Component, which is split into the Service Matchmaker Component (matching the user requirements to service functional features) and the Contract Component (matching the user requirements to contract offerings).
- The Comparison Component, which produces a report with the best option for each criterion, by facilitating grouping and ranking of the offerings.

The tool gets as input user related information such as the data location, the roles involved in the transaction, contact details of those responsible for defining the purpose of use for the involved data), contextual information on the environment setting, the user needs and requirements, the cloud service offerings in structured form, models of cloud contracts and points of attention, reputation information with respect to the agents involved in the offering process and the knowledge base of the threats and their associated mitigation control actions. The outcome of COAT is a report, including guidance on things to pay attention to, when exploring and comparing the terms of service offerings, an overview of comparable service offerings, a list of requirements that are communicated to the cloud service providers and relevant guidance for SME-specific security and privacy issues.

Through this architecture, COAT will be able to implement a set of use cases, including the support to the decision making about cloud service offerings and the understanding of the contract terms of these service offerings.

COAT offers a Web User Interface, which enables interaction with the target users. During this interaction, end users that want to be eventually cloud customers can provide as input to the graphical interface a collection of answers to a questionnaire, while the tool gives back a report and a guidance information, comparing the different offers matching the user requirements.

Furthermore, COAT offers a single interface named *Icoat,* which implements the following API methods, as shown in Table 6.

Table 6: Interfaces and respective methods provided by the Cloud Offerings Advisory Tool

Name of the API/method	*Purpose of use*	*Consumed by*	*Data format*	*API format*
Icoat/Retrieve Questionnaire	Returns a single instance of a questionnaire	The UI of COAT (target all envisaged users of the tool)	JSON	RESTful
Icoat/Manage Questionnaire	Stores, updates or deletes a single instance of a questionnaire for future reference	The UI of COAT (target all envisaged users of the tool)	JSON	RESTful
Icoat/List Questionnaires	Returns a list of all available questionnaires in the system	The UI of COAT (target all envisaged users of the tool)	JSON	RESTful
Icoat/Matches	Accepts a JSON representation of the MatchCriteria object and returns a list of matching Service offerings	The UI of COAT (target all envisaged users of the tool)	JSON	RESTful
Icoat/SOLRClient	Search for and retrieve records from a SOLR instance based on the match criteria created.	The UI of COAT (target all envisaged users of the tool)	XML /JSON implemented through SOLR structure	RESTful

Although these interface methods are currently used internally by the UI of COAT, it is examined whether they can be public so that they will be consumed by other tools as well in the future, through a web service messaging and transport layer (such as a RESTful service). A detailed analysis of the offered interface methods will be provided in a later stage.

COAT consumes the APIs provided by other tools and environments, as shown in Table 7.

Table 7: Interfaces and methods needed by the Cloud Offerings Advisory Tool

Name of the API/methods	*Purpose of use*	*Should be offered by*	*Data format*	*API format*
Countries List	Retrieve a list of countries that are available to Service Providers when adding their services. Used as options for	Cloud Providers	JSON	RESTful

	the end user to choose from where to select their location			
Service Types List	Retrieve a list of service types that are available to Service Providers when adding their services. Allows the end user to filter offers by service type when selecting their questionnaire	Cloud Providers	JSON	RESTful
Questionnaire List	Retrieve a list of Questionnaires that can be chosen by the user to complete	Questionnaire Provider	JSON	RESTful
Match	Gets details of offers that match the user's requirements based on the answers provided to the questionnaire	Questionnaire Provider	JSON	RESTful
Offer Detail	Shows a detailed view of an offer in html	Cloud Providers	JSON	RESTful

5.2 Policy Definition and Enforcement

This functional area supplements the previous one of Section 5.1 in the preventive role of the A4Cloud tools to support accountability. The set of tools belonging to this category facilitate prevention of the loss of data governance in complex cloud service provision chains through the implementation of mechanisms for the provision and support of enforceable accountability policies. Such mechanisms facilitate the definition, validation, storage, enforcement and overall management of accountability policies.

For the instantiation of the RA, two software tools are provided: AccLab (Accountability Laboratory) and A-PPL Engine (Accountable Primelife Policy Engine).

5.2.1 Accountability Lab

The purpose of the Accountability Lab (AccLab) tool is to bridge the gap between the human-readable, but abstract, accountability obligations expressed in a policy specification language, which in our case is the Abstract Accountability Language (AAL) and the concrete, but machine-readable, accountability policies expressed in Accountable Primelife Policy Language (A-PPL). AccLab provides facilities to write abstract accountability obligations in AAL via a "smart wizard" user interface and (semi-) automatically generate A-PPL policies from them to be enforced by the A-PPL Engine.

This tool is used by the privacy officers of the data controllers to express the accountability policies in an abstract form and the data subjects to express their data-handling preferences, attached to a specific policy. The tool gets as input the abstract textual definition of the policy in AAL format, expressing obligations (for a privacy officer) or the user preferences (for a data subject). The tool internally processes the obligations and identifies the mapping between these obligations to policy terms and rules, in order to generate concrete A-PPL policies for the enforcement of these obligations during the execution of cloud services. Along this process, AccLab checks the AAL statements for consistency and correctness, while compliance checking is applied to determine whether one obligation is stronger than another.

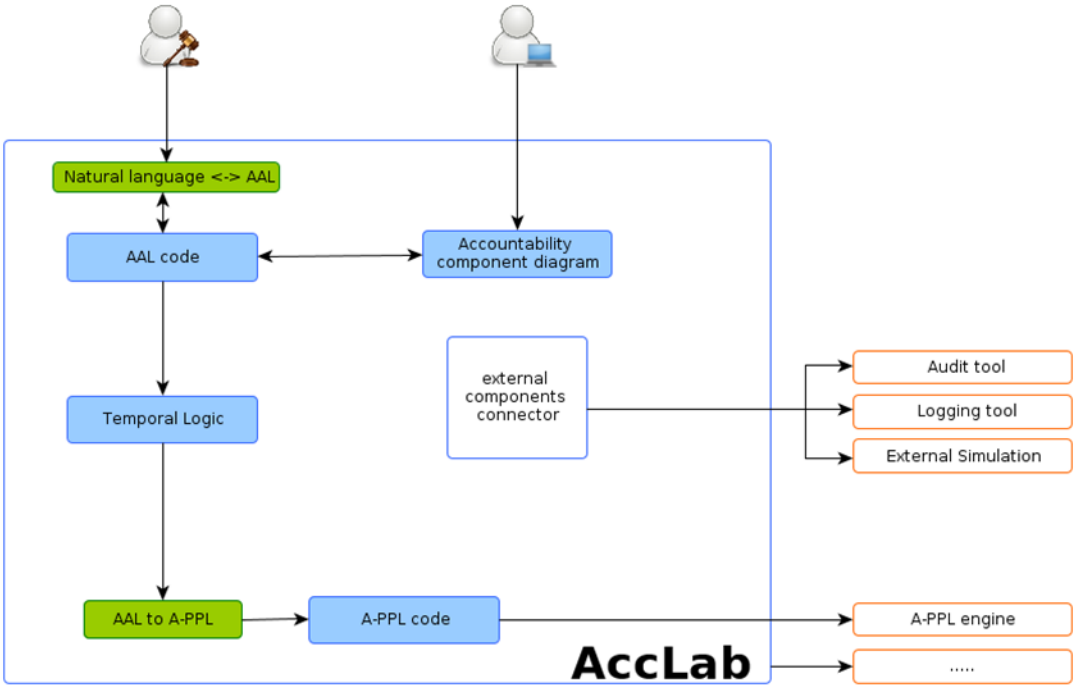

Figure 13: The high level architecture of the Accountability Laboratory tool

Figure 13 shows the high level architecture view of the AccLab. The natural obligations, i.e. textual sentences about laws, regulations and norms for data privacy preferences, are integrated with the components of a system that is designed using an accountability components diagram. The kernel of the AccLab is the AAL language (AAL code) and its semantics (Temporal logic). AAL expressions are interpreted, checked, and mapped into A-PPL code. The second part of the kernel is a connector, which can interact with a set of other tools to enable monitoring, logging, a policy engine connection, or potentially any external auditing.

Through this architecture, the AccLab produces a set of A-PPL policies (i.e. policies which conform to the A-PPL schema) and that can be fed to the policy enforcement environment.
AccLab offers a Web User Interface, which enables interaction with the target users. During this interaction, end users that want to define obligations (for privacy officers) or express privacy and security preferences (for a data subject) can use the tool and especially the UI components shown below:

- The AAL Editor, which is used to write AAL policies with editing facilities. The ALL editor targets data subjects, data controllers, auditors and data processors, who write the policies in plain AAL text and view them in HTML forms.
- The AAL Checker, which is used by data subjects, data controllers, auditors and data processors to parse and check an AAL program. This UI level functionality can also be exposed as an API, as it is shown in Table 8.
- The AAL component designer, which is used by the data controllers and the privacy officers to parse and check an AAL program (provided in JSON)

The AccLab tool offers an API relevant to the functionality exposed by the AAL Checker, as shown in Table 8.

Table 8: Interfaces provided by the Accountability Laboratory

Name of the API	*Purpose of use*	*Consumed by*	*Data format*	*API format*
AAL Parser	Parses an AAL program and performs some checks	The UI of AccLab (target all envisaged users of the tool)	JSON	RESTful

Currently, AccLab does not consume any external API by other tools and environments.

5.2.2 Accountable Primelife Policy Engine

The Accountable Primelife Policy Engine (A-PPL Engine) is an extension of the PPL engine initially designed in the PrimeLife project[9] with additional modules that enable accountability features. The main role of the original PPL engine was to enforce privacy policies related to personal data handling. A-PPL Engine extends the PPL engine with functionality that enables the enforcement of accountability obligations defined in the A-PPL language.

The tool receives the accountability policy in A-PPL format and associates the policy provisions to the piece of personal data disclosed and stored together with the data in a Personally Identifiable Information (PII) repository. This policy is referred to as a Sticky Policy in the A-PPL Engine specification. From the time that this PII is created and onwards, and for as long as a data controller holds a copy of those data, any access requests to the data are regulated by the engine, which enforces all data handling rules and obligations specified in the sticky policy. The A-PPL Engine relies on data transfer logs and evidence repositories populated by third-party tools in order to identify operations on data covered by a sticky policy that occur outside of its reach and ultimately determine whether a policy violation has occurred.

The A-PPL Engine is used by Cloud Providers who are Data Controllers and Cloud Auditors. Both roles use this tool to enforce the accountability policies, defined through an AAL policy specification tool, like the AccLab tool. Along with the A-PPL policies, the engine gets as input events (signifying trigger actions as described in obligations), relevant evidence from the environment and data access request. The tool analyses the policies by retrieving access and usage control rules and enforces the relevant obligations described in them, through analysing triggers and actions. It, also, provides configuration of handlers for events monitoring and obligations, collection of evidence and message exchanges related to event handling and action executions.

[9] http://primelife.ercim.eu/

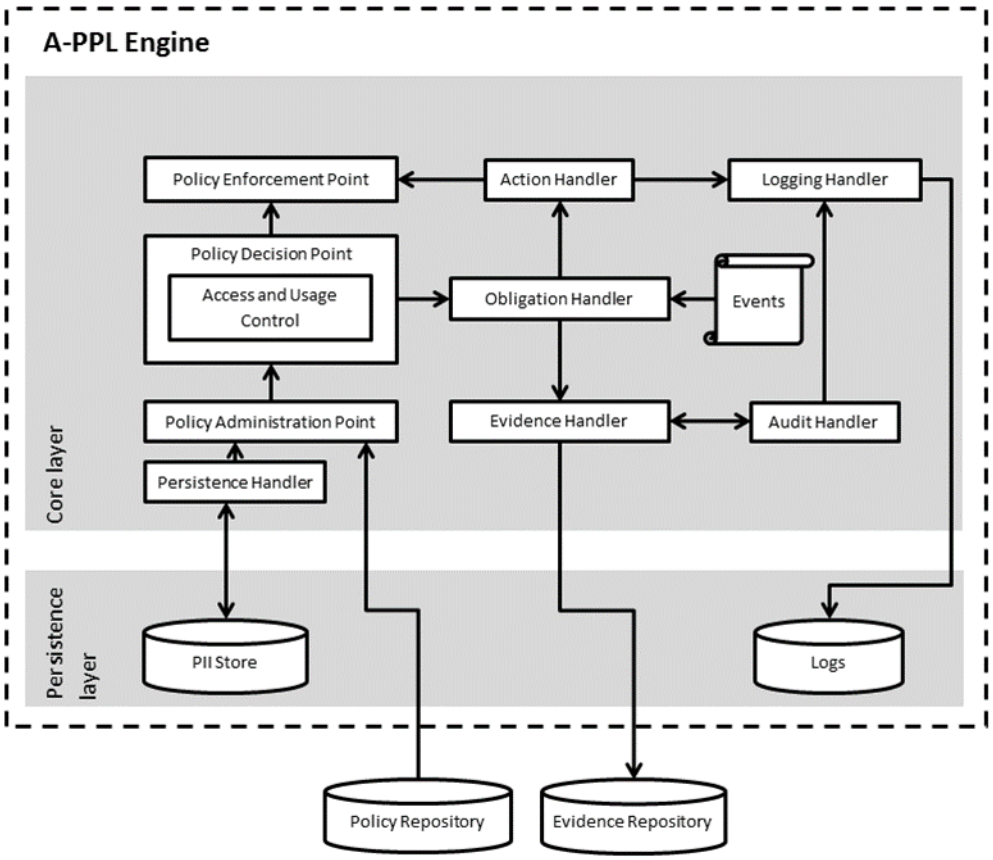

Figure 14: High level architecture of the Accountable Primelife Policy Engine

Figure 14 shows the high level view of the A-PPL Engine. As shown there, the specification of the A-PPL Engine adopts a two-layer high-level architecture to enforce isolation of engine components and data: The core elements of the policy engine providing the enforcement functionality reside in the Core layer. All personal data and their associated sticky policies stored in the PII Store reside in the Persistence layer. Access to the persistence layer is mediated through a Persistence Handler component, which abstracts the underlying location and storage data model to the core layer functions above. The architecture defines the Policy Decision Point (PDP) as the central element of the A-PPL Engine, responsible for taking access control decisions, with regards to a data access request.

More specifically, the A-PPL Engine consists of the following main components:

- The Policy Decision Point (PDP), which is responsible for taking access control decisions.
- The Policy Enforcement Point (PEP), which the PDP decisions with respect to allowing access to a piece of data.
- The Policy Administration Point (PAP), which records the obligations associated with the PIIs.
- The Obligation Handler, which executes A-PPL obligations related to the personal data handling and is complemented by the Event Handler (serving as the entry point to trigger the events which are responsible for the event based triggers) and the Action Handler (facilitating actual action execution)
- The Logging Handler, which provides an extensible secure logging interface to support all logging related functionality during the policy enforcement and personal data handling process.
- The Evidence Handler, which is responsible for the compilation of evidence, based on engine operations.
- The Audit Handler, which provides a central component for handling audit requests by facilitating the process of retrieving the necessary information from the various sub-components.

Through this architecture, the tool generates evidence that are captured and/or created by the engine itself (evidence can include logs, messages, events, etc.). An audit report can also be produced to facilitate requests from cloud auditors.

The A-PPL engine provides customised User Interfaces, which enable interaction with each of the envisaged type of users. During these interactions, end users can use a set of UI functions that are the outcome of relevant API calls offered by the A-PPL Engine and the respective *IAppEngine* Interface, as shown in Table 9.

Table 9: Interfaces and the respective methods provided by the Accountable Primelife Policy Engine

Name of the API/method	*Purpose of use*	*Consumed by*	*Data format*	*API format*
IAppEngine / storePii	Stores PII inside the PII repository, together with attached A-PPL policy that defines how the data can be processed	The UI of the A-PPL Engine (for the Data Controller)	XML-based A-PPL specification	RESTful
IAppEngine / deletePii	Deletes PII from the PII repository	The UI of the A-PPL Engine (for the Data Controller)	Key-value pair (PII id)	RESTful
IAppEngine / getPii	Retrieves specific PII given a set of attributes	The UI of the A-PPL Engine (for the Data Controller)	Key-value pair (PII id)	RESTful
IAppEngine / getAllPii	Retrieves all PII given a specific owner	The UI of the A-PPL Engine (for the Data Controller)	JSON	RESTful
IAppEngine / updatePii	Updates/corrects specific PII's value (not the policy)	The UI of the A-PPL Engine (for the Data Controller)	Key-value pair (PII id)	RESTful

A-PPL Engine consumes the APIs provided by other tools and environments, as shown in Table 10.

Table 10: Interfaces and methods needed by the Accountable Primelife Policy Engine

Name of the API	*Purpose of use*	*Should be offered by*	*Data format*	*API format*
Trigger Policy Violation	Generates a notification when PII location has changed or any other incident has been generated. All the input parameters are implicitly logged, and a potential policy violation event is triggered	Data Transfer Monitoring Tool and Audit Agent System	JSON	RESTful

5.3 Evidence and Validation

The Evidence and Validation functional area offers accountability support in both a preventive and detective manner. The set of tools belonging to this category facilitate the assertion over the prevention mechanisms against the loss of data governance in complex cloud service provision chains and the

monitoring mechanisms of the appropriate software resources to control and verify the accountability policy-based operation of these chains. More specifically, the tools in the Evidence and Validation area fulfil two main purposes:

- They provide means for the audit of cloud applications and infrastructures during their execution. This objective principally requires that appropriate auditing policies can be defined, that data be monitored during execution and evidence be stored, and that auditing properties be verified at appropriate times during execution.
- They enable the validation of A4Cloud tools. Properties to be validated include, for example, that actions of one tool do not invalidate hypotheses needed for the application of another and that information is passed correctly between tools.

The respective mechanisms to meet these objectives are being developed within the scope of the A4Cloud software tools, namely the Audit Agent System (AAS), the Data transfer Monitoring Tool (DTMT) and the Assertion Tool.

5.3.1 Audit Agent System

The Audit Agent System (AAS) tool enables the automated audit of multi-tenant and multi-layer cloud applications and cloud infrastructures for compliance with custom-defined policies, using software agents. Agents can be deployed at different cloud architectural layers (i.e., network, host, hypervisor, IaaS, PaaS, SaaS) with the purpose of i) collecting and processing evidence, ii) generating audit reports and iii) aggregating new evidence. AAS uses audit tasks that specify the data collection sources and tools to use to collect data, and policies to specify the thresholds and constraints, against which the evidence is examined to generate the audit results.

An audit component is central to any approach to accountability. Remediation actions that have to be performed after some policy has been violated, for instance, often rely on fine-grained monitoring facilities and extensive analysis capabilities of the resulting evidence. The AAS tool provides suitable means for the runtime monitoring of cloud applications and infrastructures, the verification of audit policies against the collected data, and the reporting of policy violations along with the evidence supporting it.
AAS is used by auditors, who may act on behalf of the cloud customer (external view) or the cloud provider (internal view) to perform continuous and periodic audits. The goals and nature of policies which are audited may differ depending on the view. The view may also differ in case of a trusted third-party auditor (TPA), who is independent from the customer and provider, but acting on behalf of any of those.

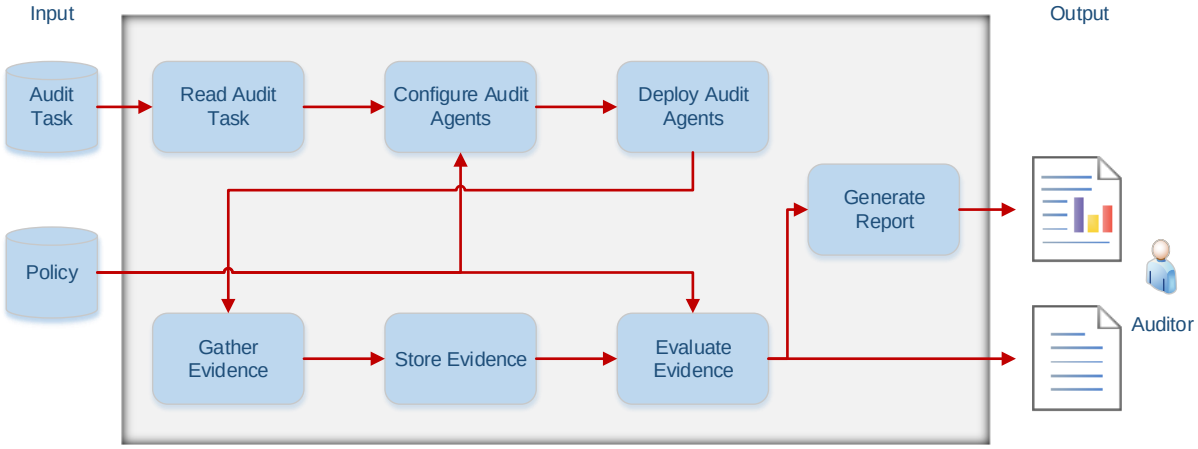

Figure 15: The high level architecture of the Audit Agent System tool

Figure 15 shows the high level view of the AAS architecture, with the following components:

- Read Audit Task: This function reads an audit task template, that is configured in the next step as input as input
- Configure Audit Agents: This function configures agents according to rules defined in the policy (rules for compliance/non-compliance) and audit task information (where and how to collect the necessary information)
- Deploy Audit Agents: This function deploys specific audit agents at necessary locations at the Cloud infrastructure

- Gather Evidence: This function represents agents that are placed at various Cloud architecture layers, where they collect evidence
- Store Evidence: This function uses stores evidence information for evaluation at a later point in time
- Evaluate Evidence: This function uses evidence information to proof policy compliance
- Generate Report: This function generates human readable reports

The tool receives as input the policy to monitor and an audit task description (describing the data collection sources and tools to use to collect data). Both these descriptions are stored in a machine-readable format. The specification of audit tasks is done by an auditor using a Web-based User Interface. A cloud provider may also provide several pre-defined audit tasks to choose from and customise.

AAS takes advantage of this input to configure the software agents specific to the collection tool, which needs to be used, and deploys them in agent runtime environments, located at different cloud architectural layers. Which agents are being deployed and therefore which data are being collected depends on the audit task. The requirements expressed as policy terms are evaluated against the collected data (which is the evidence). The evaluation results are used to generate audit reports containing audit results and supporting evidence. For evaluation of the cloud service delivery chains, intermediate results are passed out for further evaluation.

The output of the AAS tool is a report containing compliance status and supporting proof/evidence for the compliance claim. Additional output may also be (aggregated) evidence from the data gathered during the audit process (e.g., output of intermediate results for further evaluation). The report is presented to the auditor in the form of Web dashboard.

AAS offers a Web User Interface, which enables interaction with the target users. During this interaction, the auditors can use the dashboard to configure (e.g., data collection tool configuration) and control the software agents (e.g., agent lifecycle) and the audit process, as well as to check the compliance status.

The interaction of the auditors, as the end users of the AAS tool and the tool itself, is based on a set of UI functions that are the outcome of relevant API calls offered by the AAS tool, as shown in Table 11. These interfaces can be broadly categorised into being audit/AAS-specific and evidence-specific. Evidence-specific (*IEvidence* Interface) is closely related to functionality described in the Framework of Evidence (please refer to the work in WP:C-8 for more details) and the collection and storage of evidence, whereas audit/AAS-specific (*IAudit* Interface) provides an audit-focused layer on top of the Framework of Evidence functionality, which also includes the definition and evaluation of audit policies as well as the presentation of compliance results.

Table 11: Interfaces and respective API methods provided by the Audit Agent System

Name of the API/method	*Purpose of use*	*Consumed by*	*Data format*	*API format*
IAudit				
Authenticate	Authenticate the user of AAS using a suitable A4Cloud authentication backend mechanism. The visibility of audit report details may be restricted depending on the role of the user. All further calls to the public interfaces will use some kind of authentication token. Preferably, a SAML-based system will handle authentication and the release of required attributes.	The UI of the AAS (for Auditor, Cloud Provider, Cloud Customer)	-	RESTful

SetAuditPolicy	Definition of audit policy to be performed for checking compliance with A-PPL policies (it links to the next three methods setAuditTask, setPolicyThreshold and setAuditSchedule)	The UI of the AAS (for Auditor)	XML	RESTful
SetAuditTask	Definition of audit task to be performed for checking compliance with audit policy	The UI of the AAS (for Auditor)	XML	RESTful
SetPolicyThreshold	Set or update threshold values in audit policies	The UI of the AAS (for Auditor)	XML	RESTful
SetAuditSchedule	Adjusts the audit schedule for a specified audit policy (i.e., run periodically in specified intervals or continuously)	The UI of the AAS (for Auditor)	XML	RESTful
RequestAuditReport	Requests the audit results of a single or multiple audit policies	The UI of the AAS (for Auditor, Cloud Provider, Cloud Customer)	XML	RESTful
IEvidence				
RequestEvidence	Requests a record, or a chain of records respectively, from the evidence repository, given an identifier of the record	Other tools	XML	RESTful

Furthermore, AAS consumes the APIs provided by other tools and environments, as shown in Table 12.

Table 12: Interfaces and methods needed by the Audit Agent System

Name of the API	*Purpose of use*	*Should be offered by*	*Data format*	*API format*
Dispatching of policy violation alerts	Forwarding of policy violations to reach stakeholders and notify them about violation	A-PPL Engine	XML	RESTful
Requesting of A-PPL policies from policy repository	Parsing of policy and (automatic/semi-automatic) deriving of audit tasks	Either A-PPL Engine or AccLab through the policy repository	XML	RESTful

5.3.2 Data Transfer Monitoring Tool

The Data Transfer Monitoring Tool (DTMT) aims to enable cloud service providers to demonstrate compliance with personal data protection and other regulations regarding where and by whom the processing occurs. The tool automates the collection of evidence about how data transfers comply with the obligations, concerning personal data handling procedures, and are being carried out properly, by generating logs regarding the operations performed on personal data involving transfer at the infrastructure level (for example when performing load-balancing or creating backups and storing them in different hosting machines). It, then, analyses these logs using rules, helping an auditor to identify whether all transfers were compliant with the authorisations from the Data Protection Authority being obtained beforehand by the data controller. In this way, policy violations can be identified.

Part of the necessary information to monitor data transfers can be obtained from machine-readable policies specifying accountability obligations, such as A-PPL policies. Furthermore, the tool enables manual configuration for constraints about which parties, and geographical locations are allowed for a given data set.

Upon identification of a potential violation (by the tool), further data can be collected to provide supporting evidence of an incident. Notifications can be triggered by the auditor. The tool relies on the A-PPL Engine to carry out actions for policy enforcement related to the events it generates and to other tools to act upon the detected violations (e.g. for notification or remediation).

The DTMT is used by the privacy officers of data controllers and the internal and external auditors hired by cloud providers and/or customers. The tool is configured with information for the geographical location of host machines in order to determine the current physical location of the data. This information is not part of the A-PPL policy statement, but must be provided by the data processors (such as IaaS providers), so that the monitoring part of the DTMT to work properly.

The data controller needs to feed the DTMT with the authorisations made to data processors and other third parties that handle personal data. A configuration file containing these facts is filled by the data controller with information obtained from the data processors (who can further delegate the processing to other parties with the prior permission of the data controller and consent from the data subjects). Thus, if there is a transfer of the personal data to a party outside this list, it is deemed as a violation of privacy obligations. Furthermore, DTMT is fed with queries related to data location.

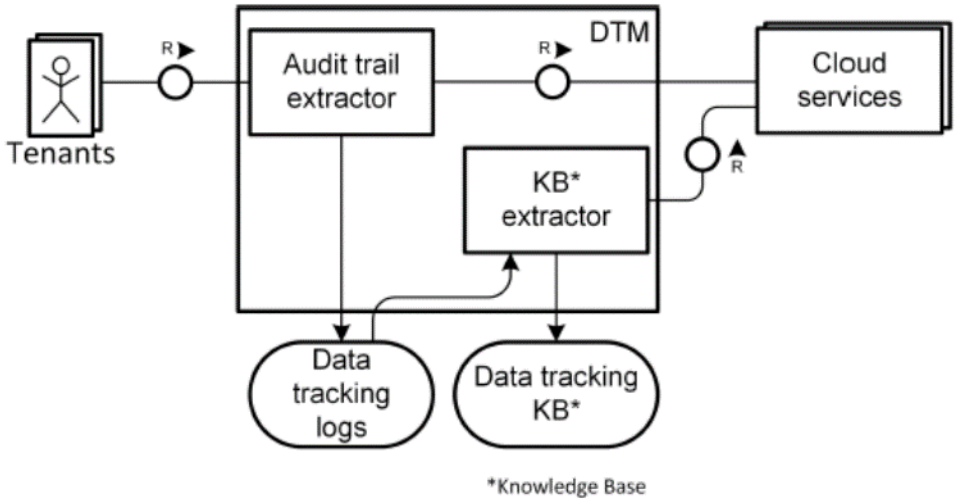

Figure 16: The high level architecture of the Data Transfer Monitoring Tool

Figure 16 shows the high level view the DTMT internal architecture. The tool continuously logs any activity on the cloud infrastructure that implies data transfers. It, then, performs inference operations on the logs to identify whether transfers were compliant to the defined A-PPL policies and/or constraints manually configured in the tool. Thus, DTMT is able to answer specific queries about the location of the processing and the parties involved in the processing for the given data set, by producing logs related to data transfers.

As shown in Figure 16, the DTMT tools is mainly composed of two main parts. Initially, a proxy module is used to monitor the machine interface calls from different tenants (e.g. data controllers or cloud platform administrators) to the cloud services and extract the audit trail. The latter is exploited to construct a data tracking knowledge base that represents operations on personal data as logical facts, suitable for analysis. The knowledge base involves an inference engine that can assert where a given data set is located in the virtualised environment. It can also answer to audit queries that help an auditor to check whether previous data transfers where compliant.

Currently, the DTMT tool is designed as a standalone tool, which is accessible through command line to run queries over the collected logs. However, the tool will be evolved to interface with the A-PPL

Engine to trigger notifications to the users, to collect logs about data transfers at the upper layer (SaaS or PaaS), and to obtain relevant information from the accountability policy in place, about the authorised transfers.

In that respect, the DTMT tool will implement the *Idtmt* Interface as shown in Table 13.

Table 13: Interface and the respective methods provided by the Data Transfer Monitoring Tool

Name of the API/method	*Purpose of use*	*Consumed by*	*Data format*	*API format*
Idtmt				
fc_check_data_location	It returns all locations where the entered personal data reference are and were stored	DTMT users and potentially by A-PPL Engine	XML	RESTful
fc_check_instances	It returns the infrastructure resources where the entered personal data reference are and were stored	DTMT users and potentially by A-PPL Engine	XML	RESTful
fc_check_processors	It returns all parties that processed a given data set	DTMT users and potentially by A-PPL Engine	XML	RESTful
fc_check_violations	It returns all unauthorized transfers (to location, or to party) detected	DTMT users and potentially by A-PPL Engine	XML	RESTful
show_authorized_transfers	This function simply shows what are the authorizations set for a data set	DTMT users and potentially by A-PPL Engine	XML	RESTful

5.3.3 Assertion Tool

The Assertion Tool (AT) is part of the Assertion Framework, a framework that provides assurance to the cloud service provider of the valid combination of the A4Cloud tools. Within the scope of the Assertion Framework, AT ensures the validation of the A4Cloud tools, using a test case-based validation methodology that tests the interactions held among two or more tools.

The assertion tool ensures the validation of the A4Cloud tools through a test case-based validation methodology. This validation might involve data coming from other tools (e.g. collecting logs through AAS). AT is responsible for gathering those data and their application on the validating scenario. Because AT is an aspect oriented programming-based tool, it could also be used to extract, analyse and transmit data that are not directly accessible from other tools.

The validation of the A4Cloud tools takes place before their delivery to the tools' users. In that respect, AT offers specific functionalities for the validation by making specific functionalities available for the A4Cloud developers, prior to the final release of the tools. This means that the AT may not be available for non A4Cloud users.

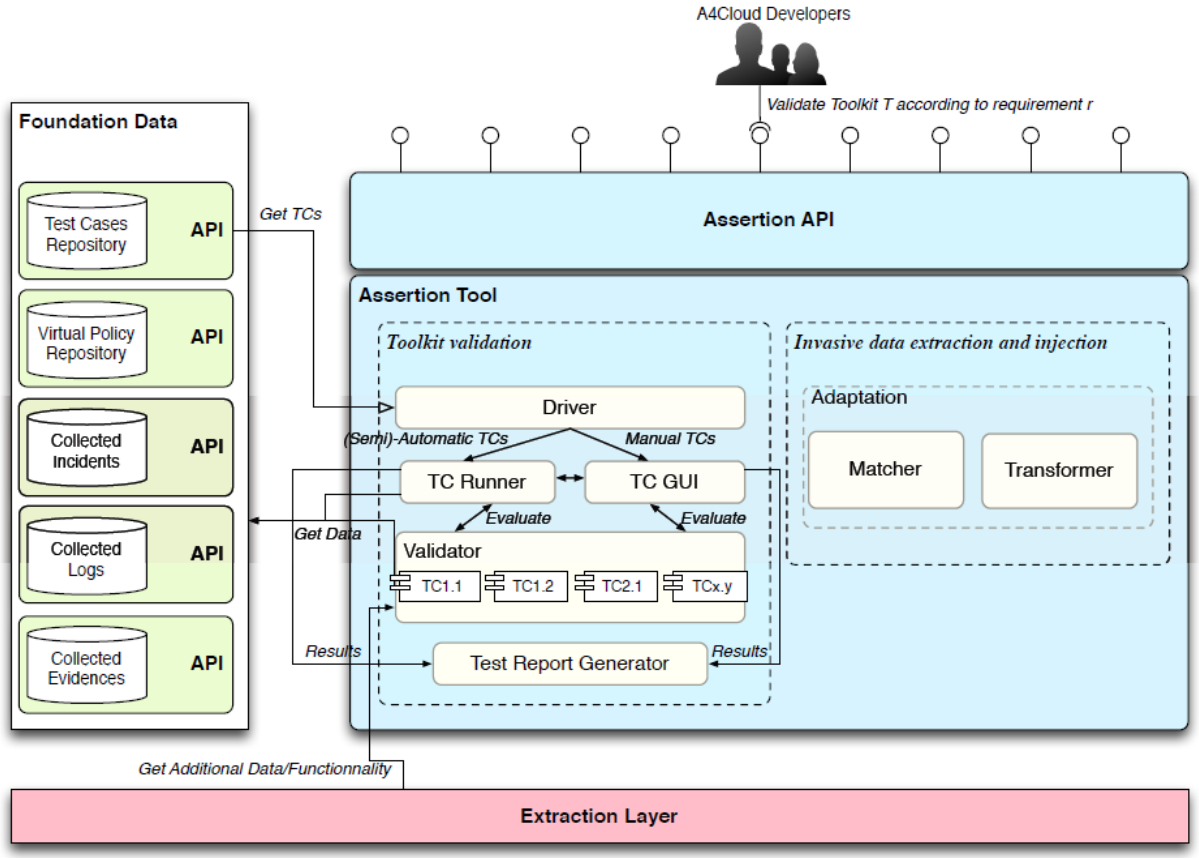

Figure 17: The high level architecture of the Assertion Tool

Figure 17 shows the high level view of the AT architecture. The tool gets as input a combination of the A4Cloud, a set of accountability requirements to be validated and a set of associated test-cases through which validation will be performed. Then, the AT automatically executes the test cases on the selected A4Cloud tools to accomplish the validation tasks. The well execution of these test cases may require direct information coming from the tools (e.g., logs, audit reports, ...) or information that might be not directly offered by the tools (e.g., timestamp whenever missing). To get this additional information AT uses Aspect Oriented Programming techniques, which enable invasive transformations at specific points in the control flow of the service application. The outcome of this tool is a report stating whether or not the tool combination is accountable. In case that the result shows that the combination is not accountable, the report contains requirements that have not been validated.

AT partially supports a Web User Interface, which enables interaction with the target users, in the sense that the developers can work with two Domain Specific Language (DSLs) for adapter definitions. The first DSL is a protocol based language for the description of (automata of) events. It enables writing queries that specify where a modification has to take place. The second DSL is used for the definition of modifications. The DSL is based on Java and enables the description of extraction of information and the modification of service calls and implementations.

Table 14 shows the APIs provided by the AT.

Table 14: Interfaces provided by the Assertion Tool

Name of the API	*Purpose of use*	*Consumed by*	*Data format*	*API format*
Assertion API	Test an Assertion for a given toolkit. Instantiates an evaluation for a given toolkit over an accountability attribute.	The UI of the AT (for A4Cloud developers)	-	RESTful

Furthermore, AT consumes the APIs provided by other tools and environments, as shown in Table 15.

Table 15: Interfaces needed by the Assertion Tool

Name of the API	*Purpose of use*	*Should be offered by*	*Data format*	*API format*
Extraction API	Operations to get or generate the logs, evidences, incidents and policies associated to a given tool	A4Cloud tools	JSON	RESTful
Foundation Data API	Allows access to data (test cases, policies, incidents, logs and evidences) needed for an A4Cloud tools validation	Ideally by the repositories provided by the A4Cloud tools	JSON	RESTful

5.4 Data Subject Controls

The Data Subject Controls functional area offers accountability support in a detective manner. The set of tools belonging to this category facilitate the need of the data subjects (i.e. individuals whose personal data are collected and/or processed by cloud service providers) to verify the correct treatment of their data. The tools implement runtime mechanisms to follow the proper execution of data handling accountability policies and provide data subjects with notifications on how their data are exploited along cloud service chains.

In the context of the A4Cloud architecture, these mechanisms are being developed within the scope of the A4Cloud software tools, namely the Data Track and the Transparency Log. We also include in this section a separate description of the Plug-in for Assessment of Policy Violation, but this plug-in is considered to be an integral part of the Data Track tool.

5.4.1 Data Track

Data Track (DT) is a tool used by data subjects to get an overview of all personal data they have disclosed. The tool allows them search through their history of data disclosures to see what personal data they have disclosed, to whom they have disclosed these data to (i.e. which data controller), and under which privacy policy. Furthermore, the DT tool enables users to directly assert their right to access and rectify the data concerning them held by the data controller. The tool can be used by both active and passive data subjects.

Specifically, DT enables data subjects to:

- Request access to their data stored at a data controller;
- Request to correct their data;
- Request that their data should be deleted or blocked;
- View detailed information about how the data has been shared and used by the data controller;
- View prior data disclosures and associated metadata like privacy policy, time, etc.;
- Detect policy violations by matching log data;
- Seek redress (via handing-over to relevant tool).

DT is consumed by data subjects, through a front end visualisation module. It gets as input the user's data disclosures, the data handling policy, describing relevant operations and obligations for the particular data disclosure, the credentials to remotely access user data at the cloud service providers' side, and the notifications received from the providers. DT uses an indexer to index and categorise all data disclosures. Upon a request from the user, DT remotely accesses the user's data located at different providers to gather the additional personal data collected by the service provider about the user. The tool can also support correcting and requesting deletion of personal data at cloud service providers. Last, but not least, DT analyses log data about how personal data has been processed and notifications to detect policy violations.

The output of the tool is a list of data disclosures for both explicitly disclosed personal data and implicitly collected data by service providers, potentially stored remotely, presented via different visualisations. Users are also presented with notifications from service providers, for example to inform them about policy violations or privacy policy updates from the service provider.

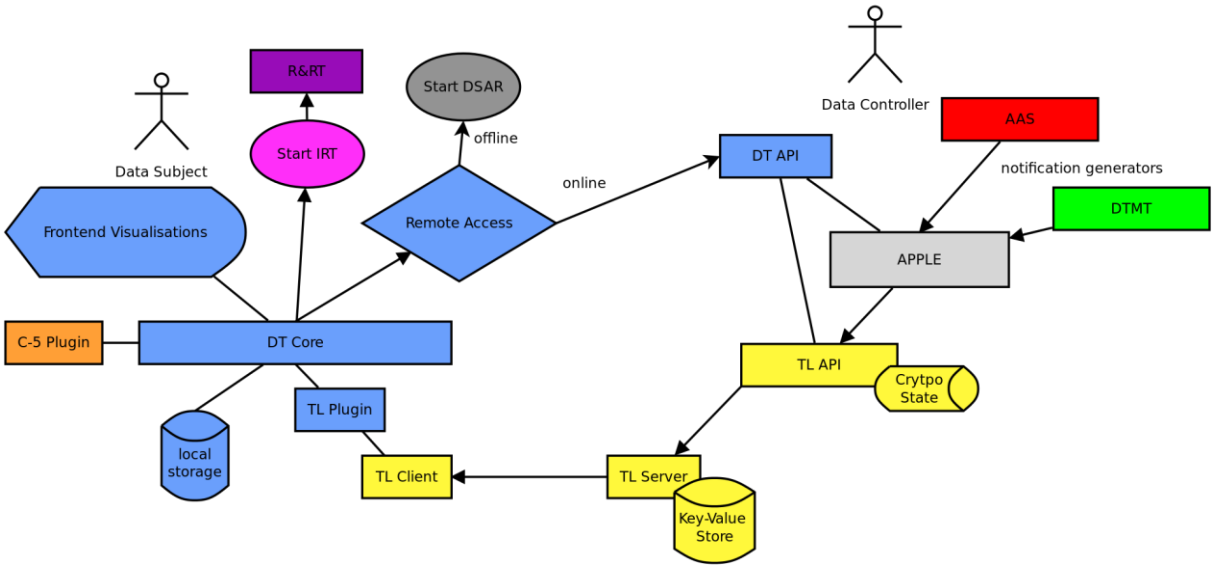

Figure 18: High level view of the Data Track and Transparency tools

Figure 18 illustrates the DT tool and the interactions with other A4Cloud tools. The tool consists of the blue highlighted functional boxes. The core component of the Data Track is the DT Core. The DT Core acts as a backend with local storage and support for plugins that get data from remote sources. The core provides access to data from its plugins in a normalised form to the Frontend Visualisations. An example of such a plug-in is shown in Figure 18 for the interaction with the Transparency Log (TL) tool, which provides notifications from the A-PPL engine (and potentially also logging data) to DT. Possible generators of notifications are the AAS and the DTMT. It must be noted that we consider to integrate the data track logic assigned to the data controller into the A-PPL Engine, so that DT is only installed on the data subject side.
In case of notifications concerning incidents or violations, the DT Core is using a policy violation plug-in to assess the severity of these notifications. The severity will impact how the notifications are presented to the data subject by the Frontend Visualisations component. For example, a high severity may cause a notification to the data subject as soon as the tool is launched, while less severe notifications are only shown on request. If the data subject wishes to act on a notification concerning an incident or a violation, the DT Core can launch the Incident Response Tool.

If the data subject wishes to perform actions that involve remote access to a cloud service provider (i.e. a data controller), there are two cases:

- In case of online access being possible, the DT Core uses the *DT API* component running at the data controller. The DT API interacts with A-PPL Engine (and potentially other components at the data controller) to accomplish the remote access request.
- When online case is not possible, the DT Core can start the Data Subject Access Request tool.

DT offers a Web User Interface to serve the front end visualisation module. This UI provides different visualisations of the user's data disclosures. One of the visualisations shows traces indicating what attributes a user has disclosed to which providers, by visually linking cloud service providers and attributes. Another visualisation shows a timeline of the user's data disclosures. The users will have the possibility to filter out displayed services or attributes.

Furthermore, DT provides a set of API methods that are offered by the *Idatatrack* Interface and can be consumed both by the UI of the DT and other A4Cloud tools. These interface methods provided by DT are shown in Table 16.

Table 16: Interfaces and respective methods provided by the Data Track tool

Name of the API	*Purpose of use*	*Consumed by*	*Data format*	*API format*
Idatatrack / SetupTL	Setup the data subject at the Transparency Log	The UI of the DT (for data subjects)	JSON	RESTful
Idatatrack / Authenticate	Authenticate a data subject using the Transparency Log	The UI of the DT (for data subjects)	JSON	RESTful
Idatatrack / GetPersonalData	Get all data associated to the authenticated data subject	The UI of the DT (for data subjects)	JSON	RESTful
Idatatrack / RequestDataCorrection	Requests that a provided attribute should be corrected (changed) to the provided value	The UI of the DT (for data subjects)	JSON	RESTful
Idatatrack / RequestDataDeletion	Requests that all data associated to the authenticated data subject should be deleted	The UI of the DT (for data subjects)	JSON	RESTful

In order to better understand the APIs exposed by the DT tool and since this tool is strongly coupled with the TL tool, we present the following scenario to explain the use of the APIs. For an active data subject, the data subject generates his or her own asymmetric key-pair. For a passive data subject, the data controller generates the key and stores all key material for the data subject (with the obvious loss of for example confidentiality if you do not trust the data controller). Regardless, we assume the following:

- The data subject has a public key, as defined by Curve25519[10].
- The data controller has an identifier for the data subject. This may be an account (like alice@example.com) or a transaction pseudonym (like a session cookie).

Given a data subject's public key and an identifier, the data controller uses the API method *SetupTL* to setup the Transparency Log. In case of an active data subject, the reply is returned directly to the data subject. In case of a passive data subject, the reply is stored by the data controller. After setup, the active data subject (or passive data subject that retrieved their key material) has the possibility to authenticate using the API method *Authenticate*. In A4Cloud, we use the TL key to authenticate. The reply from a successful *Authenticate* is an authentication key used to authenticate calls to the other API methods. The authenticated API methods are:

- *GetPersonalData* – returns all data associated with the authenticated data subject. Associated metadata should describe the data.
- *RequestDataCorrection* – requests that a provided attribute should be corrected (changed) to the provided value.
- *RequestDataDeletion* –requests that all data associated with the authenticated data subject should be deleted.

Furthermore, DT consumes the APIs provided by other tools and environments, as shown in Table 17.

Table 17: Interfaces and methods needed by the Data Track tool

Name of the API	*Purpose of use*	*Should be offered by*	*Data format*	*API format*
TL Client	Retrieve all data sent to the data subject through the Transparency Log. Also provides access to other Transparency Log functionality such as cryptographic proofs of	Transparency Log	Plain text for data (that is, retrieved data is in the same format as written),	RESTful

[10] http://cr.yp.to/ecdh.html

	authenticity, time, inconsistency, etc.		JSON or Google Protocol Buffers for cryptographic proofs	
Start IRT	Enable a data subject to start the Incident Response Tool for assistance with responding to a particular incident	Incident Response Tool	-	RESTful
Policy Violation Plugin	Enable the data subject to assess the severity of one or more incidents	Plug-in for Assessment of Policy Violation	-	Might be integrated with DT
TL API	Enable the DT API to interact with the Transparency Log to authenticate data subjects	Transparency Log	Google Protocol Buffers (internal API between DT and TL), that is base64 encoded in JSON (to fit the generic external DT API)	RESTful
A-PPL API	Enable the DT API to interact with A-PPL Engine for API calls GetPersonalData, RequestDataCorrection, and RequestDataDeletion	A-PPL Engine	-	RESTful

5.4.2 Plug-in for Assessment of Policy Violation

The Plug-in for Assessment of Policy Violation (PAPV) provides an assessment on the relevance of previously detected policy violations. By using it, data subjects (or their representatives) can check which policy violations are the most relevant ones to the data subject itself.

PAPV is used by data subjects to assess policy violations. The plug-in gets as input a collection of instances of the policy violations from the DT tool, where an instance of a policy violation is any piece of evidence that describes an occurrence of a policy violation event, and a machine-readable policy description, detailing the obligations of the data controller, regarding the data handling procedures treatment for the personal data of the data subjects, which will serve as a reference for evaluating the detected violations. The plug-in can also be fed with a document describing the data subject's preferences with respect to the treatment of their data, which will guide the assessment.

By receiving a list of the latest policy violations, together with their associated policies, PAPV produces an assessment for the relevance of each reported violation. It, then, prepares an ordered list of policy violations, sorts them by the level of importance. The output of the plug-in is an ordered measurement (either qualitative or qualitative) of the relevance of the violation event, for each instance of the policy violation. This enables the list of policy violations to be sorted with respect to their relevance.

Figure 19 shows the high level architecture of PAPV.

PAPV does not support any UI, but the functionalities provided by it are realised through the DT UI.

The plug-in will be integrated with the DT tool development and there is no need to consume any API provided by other tools. However, the API shown in Table 18 will be offered by PAPV.

Table 18: Interfaces provided by the plug-in for Assessment of Policy Violation

Name of the API	*Purpose of use*	*Consumed by*	*Data format*	*API format*
IPapv	Asses the relevance of the provided policy violation	Data Track	XML	Go library

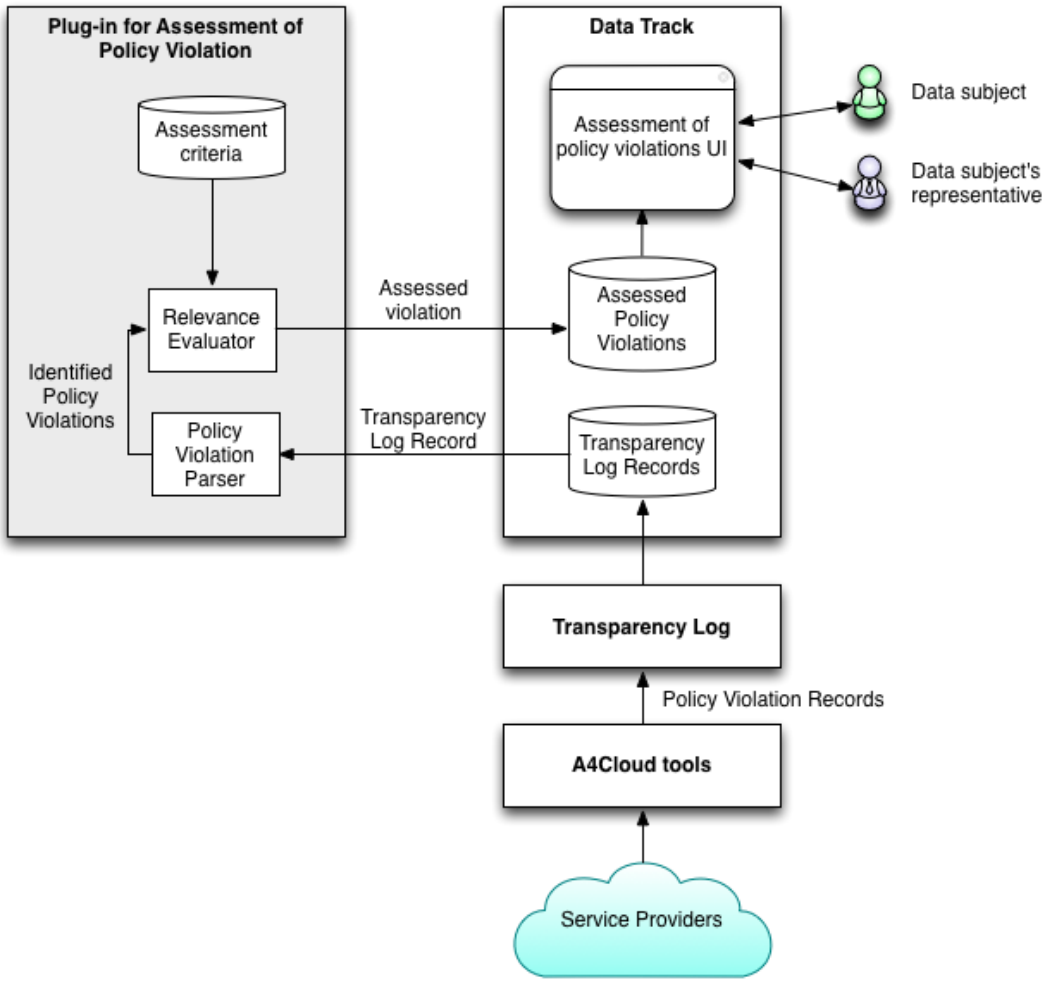

Figure 19: High level architecture of the plug-in for Assessment of Policy Violation

5.4.3 Transparency Log

The Transparency Log (TL) is based on the cryptographic scheme Insynd [23], which is used for secure and privacy-preserving unidirectional asynchronous communication in settings, where intermediate servers are considered active adversaries. TL uses Insynd to provide a one-way communication channel between service providers and data subjects. This enables a reliable channel for sending notifications to data subjects (who cannot always be assumed to be online to receive data) even for privacy-sensitive data that normally cannot be sent by email, for example.

By using this cryptographic scheme, TL has the following properties:

- Secrecy: all stored data are encrypted and can only be decrypted by the recipient.
- Fully untrusted outsourced storage: except from availability (i.e. the storage provider can deny service), an untrusted party can store all data without loss of any of the properties provided by TL.

- Forward integrity and deletion detection: in case that the service provider sends data to data subjects through TL and this provider becomes compromised, then an attacker cannot modify those data sent using TL prior to the provider being compromised.
- Forward unlinkability of events and recipients: A third party cannot tell which recipient, out of all possible data subject recipients, received what data (stored in events) from the sender.
- Publicly verifiable proofs (i.e. that any third party can verify) of:
    - The time an event (or data inside) was sent, as certified by a time stamping authority.
    - Who the recipient of a particular event is.
    - Who the sender of a particular event is.
    - The consistency of all data stored at the untrusted party.
- Support for distributed settings, where the ability to send data to a data subject (or any recipient) can travel with the data dynamically.

TL is used by data subjects, whose personal data are being processed by cloud service providers. The tool gets as input any data a service provider wishes to send to specific data subjects. TL, in turn, provides a number of cryptographic operations on the provided input, as part of a cryptographic scheme, which, among other things, prevent information leaks and anyone from modifying the data. The outcome of this tool is the data sent by the cloud service provider together with cryptographic proofs of correctness.

As presented in section 5.4.1, the TL tool is tightly coupled with DT. Figure 18 illustrates the interactions of this tool with other A4Cloud tools, in which the TL components are coloured yellow. This summarises the intended role of TL for DT. TL is, however, a more general solution consisting of three components:

- The first is the *sender*, depicted in the diagram as the TL API. The sender holds some crypto state to be able to send data.
- The second is the *untrusted server*, depicted in the diagram as the TL Server. The server provides storage of data sent by the sender to recipients.
- The third is a *recipient*, depicted in the diagram as the TL Client. The recipient receives data from the sender by querying the server.

As a cryptographic scheme, the TL tool does not provide any Web User Interface.

The tool provides a set of APIs to communicate mainly with the data controllers. Thus, the public APIs defined here mainly refer to the sender component, while the API between sender, untrusted server, and recipient remain internal to the TL tool.

TL provides the *Itlog* API shown in Table 16, in which the methods for GET, POST, and DEL do the expected operations on logs and recipients. TL does not need any interfaces provided by other tools.

Table 19: Interfaces and respective API methods provided by the Transparency Log tool

Name of the API/methods	*Purpose of use*	*Consumed by*	*Data format*	*API format*
Itlog / ListLogs	List all logs	DT (for Data Controllers)	JSON	RESTful
Itlog / SetupLog	Setup a new log to log data for recipients	DT (for Data Controllers)	JSON	RESTful
Itlog / CloseLog	Close a log, preventing further data from being logged	DT (for Data Controllers)	JSON	RESTful
Itlog / ListRecipients	List all valid recipients setup for a log	DT (for Data Controllers)	JSON	RESTful
Itlog / SetupRecipient	Setup a recipient for a log	DT (for Data Controllers)	JSON	RESTful
Itlog / RemoveRecipient	Remove a recipient for a log, preventing further data from being logged for the recipient in question	DT (for Data Controllers)	JSON	RESTful

Itlog / Log	Record data for a particular recipient in a particular log	DT (for Data Controllers)	JSON	RESTful
Itlog / GetState	Enables a recipient to query its state at the sender	DT (for Data Controllers)	JSON	RESTful

5.5 Incident Response and Remediation

The Incident Response and Remediation functional area offers accountability support in a corrective manner. The set of tools belonging to this category facilitate the need for providing mechanisms for remediation of accountability failures and incident response.

In the context of the A4Cloud architecture, these mechanisms are being developed within the scope of the A4Cloud software tools, namely the Remediation and Redress Tool and the Incident Response Tool.

5.5.1 Remediation and Redress Tool

The Remediation and Redress Tool (RRT) aims to assist individual or small SME cloud customers in responding to (perceived) incidents in their cloud arrangement. It will be activated as a result of certain incidents reported by the Incident Response Tool (see Section 5.5.2 about IRT) or can be invoked by the user on the basis of information coming from other sources (such as newspaper reports).

If the tool is triggered by the IRT, then RRT knows what type of incident has occurred and what possible actions can be undertaken. Then, it will guide the user through these actions, which may involve composing requests/questions to the cloud provider or DPAs to take action, formal complaints or other legal actions. In case the tool is consulted by the user without being triggered by the IRT, it will engage in a dialogue with the user to establish their concern and next guide the user through appropriate actions. Where appropriate, the tool will take automatic action (through potential APIs provided by cloud service providers and/or the DPAs) to communicate requests/complaints etc.

RRT gets as input the incident data (in the form of type of incident, time and scope), some user related information (e.g. about the location, the allocated roles, the contact details, etc.), some information about the cloud service provider (e.g. about the location, the allocated roles, the contact details, etc.), any contextual information that can assist in making proper decisions on remediation and the incident response model retrieved from a knowledge base.

The tool integrates mechanisms for rendering user concerns, analysing the incoming information and providing explanation, mapping the concern or incident to the knowledge base, synthesizing the response and producing the relevant report. The output of the tool is a list of potential actions and completed (standard) forms for complaints/request etc. It also educates the user on incidents and stakeholders for potential actions and procedures.

The respective functionalities are offered through the components depicted in Figure 20. This figure integrates the view of both the RRT and the IRT.

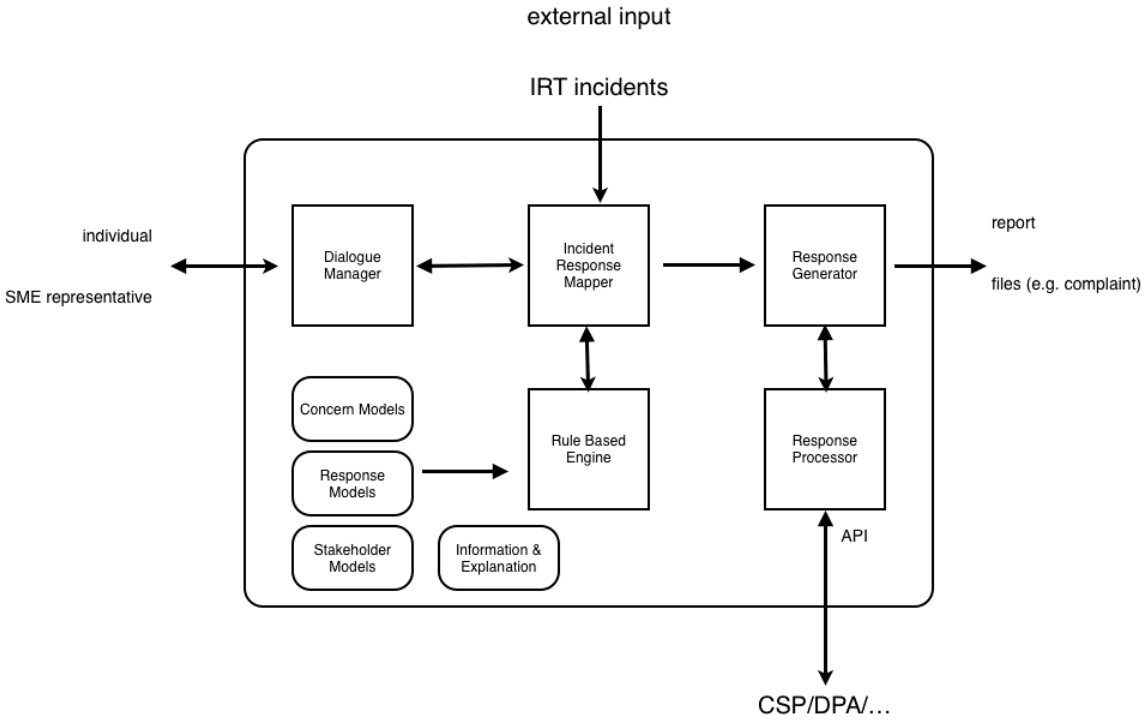

Figure 20: The high level architecture of the Incident Response Tool and the Remediation and Redress Tool

RRT offers a Web User Interface, which integrates the cases planned to be addressed by the tool, thus composing a remediation request upon invocation from the IRT and preparing the remediation request when the end use manually wants to submit one.

Currently, the tool does not consume any interface offered by other tools. As shown in Table 20, the tool offers the *IRrt* Interface, which enables interaction with the IRT to implement remediation.

Table 20: Interfaces and respective methods provided by the Remediation and Redress Tool

Name of the API	*Purpose of use*	*Consumed by*	*Data format*	*API format*
IRrt / prepareRequest	Invokes RRT to produce the remediation request, when an incident is identified	IRT	-	RESTful

5.5.2 Incident Response Tool

The Incident Response Tool (IRT) is the entry point for handling anomalies and detected violations in cloud environment scenarios, such as privacy violations or security breaches. The tool receives incident signals and takes the initial steps to respond to these incidents, by sending alerts to the user when a relevant incident has occurred based on different parameters. More specifically, IRT takes input from the evidence tools (see section 5.3) that detect incidents and filters them so that they can be reported and presented to the user in a comprehensive way. The tool, also, provides appropriate options to respond to the incident through the RRT.

The tool relies on incidents as they are detected at the cloud provider side. Incidents that can occur at the providers' side include a wide range of phenomena, such as hardware failure, data breach due to hacking, policy infringements, interception/surveillance by security agencies, use of data in breach of established policies, etc. Not all incidents can be detected automatically. For instance, it may not be possible to detect that a systems administrator has made a copy of the data in their system and sold it to an outsider. These types of incidents will (obviously) not be handled by the IRT.

Of the incidents that can be reported, some, but not all, may need to be reported to entities outside the cloud service provider (that is Cloud Customer, Cloud Subject, Cloud Supervisory Authority). The tool filters out the incidents that potentially need to be reported to the tool's user.

IRT targets individual cloud customers and SMEs. The high level view of the IRT architecture is depicted in Figure 20.

IRT develops a Web User Interface, which is used by cloud customers and data subjects to receive alerts with respect to notifications. In order to do so, IRT needs to interface with other A4Cloud tools and receive information about policy violations and any incident happening in the system, along with potential evidence on the specific incident.

In that respect, IRT offers the *IIrt* Interface, which delivers the methods shown in Table 21. Currently, IRT does not need any external interfaces, offered by other tools.

Table 21: Interfaces and respective methods provided by the Incident Response Tool

Name of the API	*Purpose of use*	*Consumed by*	*Data format*	*API format*
IIrt / getIncident	Receive a set of anomalies and violations	A-PPL Engine	-	RESTful
IIrt / sendAlert	Communicates alerts	The UI of IRT	-	RESTful

5.6 Summary of the tool interactions

After presenting the tools and their architectural design, in this section we make an attempt to allocate the A4Cloud tools to the cloud and data protections roles, as they were described in Section 2.1 and the relevant aspects of the accountability-enabled cloud service value chain, as introduced in the different paragraphs of Section 4.

Figure 21 shows the interactions between the A4Cloud tools to accomplish the functions allocated to the four interaction paths identified in the accountability practices (as they have been introduced in WP:C-2 and WP:C-3), namely agreement, reporting, demonstration and remediation. For each component, a respective interaction is shown by projecting the offered and required interfaces, as well as any potential communication with user interfaces.

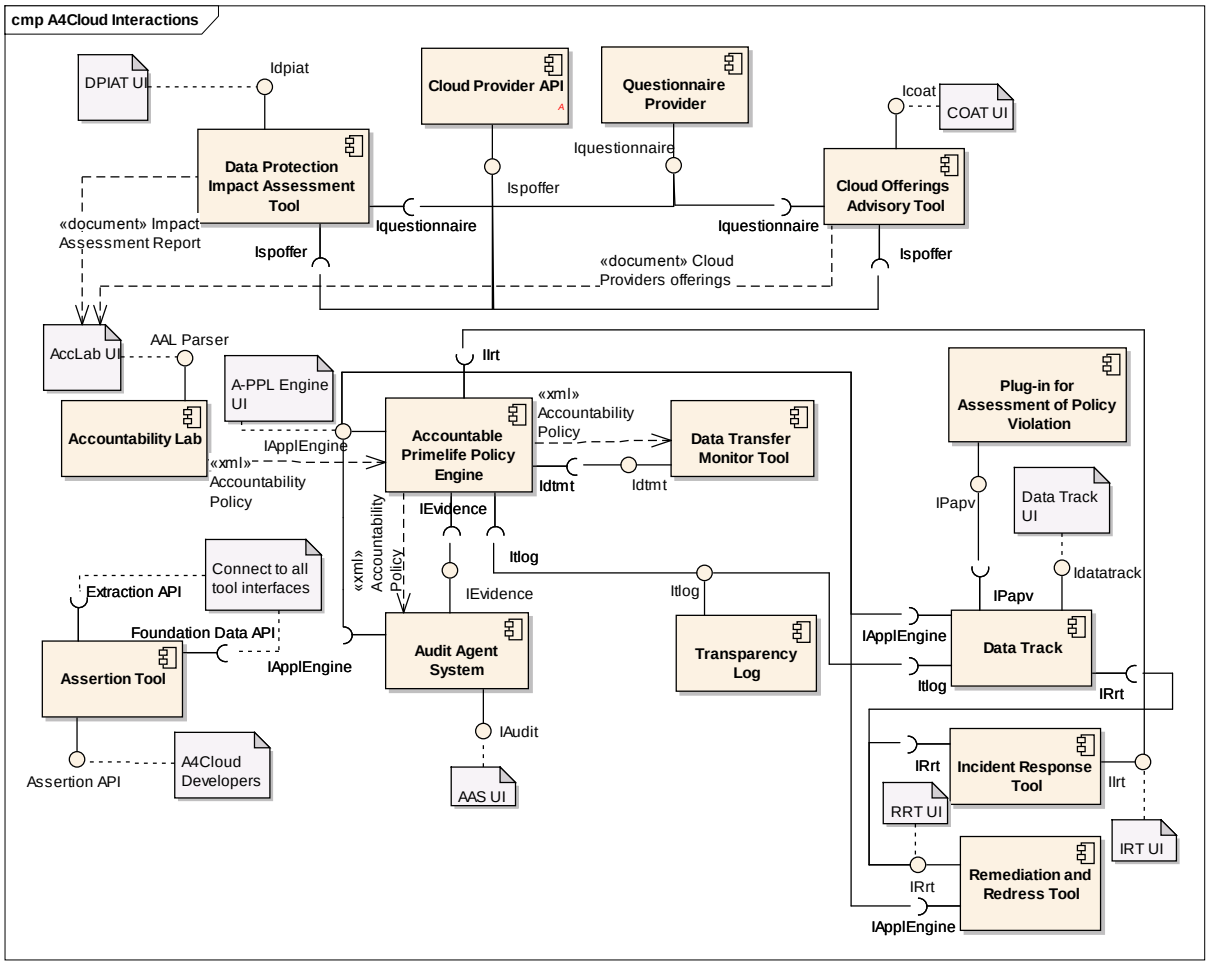

Figure 21: The interaction diagram of the A4Cloud tools

In the following, we allocate the tools and their user interfaces to the roles of Section 2.1, excluding the AT, which is used by A4Cloud developers to validate the proper operation of the A4Cloud tools. This is depicted in Table 22, in which we analyse which tools are used by each cloud role and the relevant data protection role. The analysis considers both the user interfaces of the tools that are accessible by the indicated roles and the backend tool software that needs to be deployed on the specific role machine to provide the associated accountability function. It must be noted that, compared to the list of cloud roles shown in Section 2.1, we excluded the cloud carriers and brokers, who can mainly act as data processors (and in some case as controllers), since they are attributed in the same situation as for cloud providers.

Table 22: The tools and their user interfaces used by the A4Cloud cloud and data protection roles

Extended NIST cloud role / Data Protection Role	Which User Interface Tool is used	Which Tool is hosted at this role's side	Main Tool Input	Main Tool Output
Cloud Subject / Data Subject	UI of the business use case	AccLab	Privacy and security preferences	A-PPL policy
	DT	DT	Personal data	List of data disclosures, notifications from providers
	DT	PAPV	Notification	Ranked Policy violation
	DT	TL	Data recipient	Encrypted Logs

Extended NIST cloud role / Data Protection Role	Which User Interface Tool is used	Which Tool is hosted at this role's side	Main Tool Input	Main Tool Output
	RRT	RRT	Notification	Completed complaints form
	IRT	IRT	Incident	Notification
Cloud Customer / Data Controller	DPIAT	DPIAT	Data and processes in a transaction, risk and trust models	Impact Assessment Report
	COAT	COAT	Privacy and security requirements	Report on offerings
	AccLab	AccLab	Obligations	A-PPL policy
	A-PPL Engine	A-PPL Engine	A-PPL policy	Logs
	AAS	AAS	A-PPL policy, Audit Tasks	Audit Report, Evidence
	A-PPL Engine	TL	Data recipient	Encrypted Logs
	RRT	RRT	Notification	Completed complaints form
	IRT	IRT	Incident	Notification
Cloud Customer / Data Processor	AccLab	AccLab	Obligations	A-PPL policy
	A-PPL Engine	A-PPL Engine	A-PPL policy	Logs
	RRT	RRT	Notification	Set of actions
	IRT	IRT	Incident	Notification
Cloud Provider / Data Controller	AAS	AAS	A-PPL policy, Audit Tasks	Audit Report, Evidence
	AccLab	AccLab	Obligations	A-PPL policy
	A-PPL Engine	A-PPL Engine	A-PPL policy	Logs
	A-PPL Engine	TL	Data recipient	Encrypted Logs
	AAS	AAS	A-PPL policy, Audit Tasks	Audit Report, Evidence
	RRT	RRT	Notification	Set of actions
	IRT	IRT	Incident	Notification
Cloud Provider / Data Processor	-	DTMT	A-PPL policy	Data Transfer Logs, Notifications
	AAS	AAS	A-PPL policy, Audit Tasks	Audit Report, Evidence
	RRT	RRT	Notification	Set of actions
	IRT	IRT	Incident	Notification
Cloud Auditor	AAS	AAS	A-PPL policy, Audit Tasks	Audit Report, Evidence
Cloud supervisory authority / Supervisory authority (DPA or NRA)	AAS	AAS	A-PPL policy, Audit Tasks	Audit Report, Evidence

From Table 22 and the categorization of the tools to functional areas, as done in the opening paragraphs of section 5, we also show connection of the accountability information and practices with the tools. As such:

- The policy definition and enforcement relates to the functionalities and the accountability information produced by the tools of the respective Policy Definition and Enforcement functional area. These tools handle the requirements, which can be set and/or calibrated through the tools of the Contract and Risk Management functional area, for implementing the preventive accountability mechanisms and can be used by the cloud roles to set the conditions, under which a cloud service that involves the processing of personal and/or business confidential data is operating.
- The account is progressively built by exploiting the logs collected from the cloud providers and the A4Cloud tools, namely the A-PPL Engine, AAS and DTMT. The log sources and types relate to the

tools of the Evidence and Validation functional area, which implement the respective detective accountability mechanisms. All these logs are analysed, based on the Framework of Evidence, as it is defined in WP:C-8, and they build the evidence records. The latter are used as part of an account to facilitate the reporting and compliance demonstration functions of the accountability framework.

- The analysis of the logs processed to constitute the evidence and the relevant account are used to generate incidents and notifications that drive the corresponding remediation as part of the functionalities offered by the tools of the Incident Response and Remediation functional area, in the context of the corrective accountability mechanisms.
- At all stages, the data subject functionalities are being implemented through the tools of the Data Subject Controls area.

6 Conclusion

This document comprises an intermediate step towards the delivery of the full A4Cloud reference architecture (due month 30 of the project's run) structured to provide a high-level view of the architecture for accountability we are developing. It presents the conceptual basis for the development of the architecture, our view of how accountability governance should be designed, the challenges of achieving accountability across a complex supply chain and a solution based on accountability-support services to overcome them, before focusing on the design and architecture of the tools comprising the A4Cloud toolset as concrete contributions of the project.

Moving forward this document will evolve to describe a complete reference architecture for accountability containing the following elements:

- A formal model;
- A methodology for applying, from the organization's perspective, an accountability-driven governance approach;
- A set of accountability-support services and their respective components
- A description of the externally-developed accountability-relevant tools which are used by the project
- A description of the interactions (data exchanges) between the tools and with a ruining system
- A series of patterns describing, in a generic or prototype manner, the integration of the tools with a running system
- A sample deployment architecture describing the integration of all the tools with a running system

7 References

[1] P. Krutchen, The Rational Unified Process: An Introduction, Reading : Addison-Wesley, 2000.

[2] V. Tountopoulos and et al, “Architecture guidelines and principles (internal report),” A4Cloud project, 2013.

[3] S. Pearson, M. Felici and et al., “WP-32 Conceptual Framework,” A4Cloud project, 2014.

[4] F. Liu and et al, “NIST Cloud Computing Reference Architecture,” NIST Special Publication 500-292, 2011.

[5] CIPL - Galway Project, “Data Protection Accountability: The Essential Elements,” 2009.

[6] Office of the Information and Privacy Commissioner of Alberta; Office of the Privacy Commissioner of Canada; Office of the Information and Privacy Commissioner for British Colombia, “Getting Accountability Right with a Privacy,” 2012.

[7] European Commission, “Proposal for a regulation of the European Parliament and of the Council on the protection of individuals with regard to the processing of personal data and on the free movement of such data (General Data Protection Regulation),” 2012.

[8] CNIL, “Recommendationds for companies planning to use cloud services,” 2012.

[9] Information Commissioner's Office, “Guidance on the use of cloud computing,” 2012.

[10] Nymity Inc., “Privacy Management Accountability Framework,” 2014.

[11] IT Governance Institute, “COBIT: Control Objectives for Information and related Technology,” 2000.

[12] ISO/IEC, “ISO/IEC 27001:2013: Information technology — Security techniques — Information security management systems — Requirements,” 2013.

[13] J. De Clerq and et al, The HP Security Handbook, Hewlett Packard, 2008.

[14] UK Information Commissioner's Office, “Guidance on the use of cloud computing,” 2012.

[15] J. Luna and D. Cattedu, “Report on A4Cloud contribution to standards,” A4Cloud project, 2014.

[16] A. Pannetrat and et al, “The interoperability of A4Cloud Framework,” A4cloud project, 2014.

[17] ISO/IEC/IEEE, “ISO/IEC/IEEE 29119 Software and systems engineering - Software testing,” 2013.

[18] P. Mell and T. Grance, “The NIST Definition of Cloud Computing,” NIST Special Publication 800-145, 2011.

[19] W. Benghabrit and et al, “A cloud accountability policy representation framework,” A4Cloud project, 2014.

[20] W. Benghabrit and et al, “Abstract Accountability Language,” in *8th IFIP WG 11.11 International Conference on Trust Management*, Singapore, 2014.

[21] M. Azraoui and et al, “A-PPL: An Accountability Policy Language for Cloud Computing,” in *DPM / SETOP*, Wroclaw (Poland), 2014.

[22] T. Wlodarczyk and et al, “D:C-8.1 Framework of Evidence,” A4Cloud project, 2014.

[23] T. Pulls and et al, “Distributed privacy-preserving transparency logging,” in *Proceedings of the 12th annual {ACM} Workshop on Privacy in the Electronic Society*, Berlin, Germany, 2013.

[24] K. Bernsmed and et al., “MSB-3.1 Use Case Descriptions,” A4Cloud project, 2014.

8 Appendices

8.1 List of Obligations

The following list of obligations was extracted from the WP B-3 MSB-3.1 report [24]. We point to that report for a full list of those obligations that provides extended details, including legal perspectives.

List of obligations from the regulatory perspective (Data Protection Directive), to which Cloud actors must adhere:

- **Obligation 1: informing about processing**. Data subjects have the right to know that their personal data is being processed.
- **Obligation 2: informing about purpose.** Data subjects also have the right to know why their personal data is being processed.
- **Obligation 3: informing about recipients.** Data subjects have the right to know who will process their personal data.
- **Obligation 4: informing about rights.** Data subjects have the right to know their rights in relation to the processing of their personal data.
- **Obligation 5: data collection purposes.** Personal data must be collected for specific, explicit and legitimate purposes and not further processed in a way incompatible with those purposes.
- **Obligation 6: the right to access, correct and delete personal data.** Data subjects have the right to access, correct and delete personal data that have been collected about them.
- **Obligation 7: data storage period.** Personal data must be kept in a form that permits identification of data subjects for no longer than is necessary for the purpose for which they were collected.
- **Obligation 8: security and privacy measures.** Controllers are responsible to the data subjects for the implementation of appropriate technical and organizational security measures.
- **Obligation 9: rules for data processing by provider.** Controllers are accountable to data subjects for how sub-providers process their personal data.
- **Obligation 10: rules for data processing by sub-providers.** The controller must also ensure that all sub-providers involved in the service delivery chain do not process the personal data, except on the controller's instructions (unless they are required to do so by law).
- **Obligation 11: provider safeguards.** Controllers are accountable to data subjects for choosing data processors that can provide sufficient safeguards concerning technical security and organizational measures.
- **Obligation 12: sub-provider safeguards.** The previous obligation comprises all processors in a service delivery chain.
- **Obligation 13: informed consent to processing.** Controllers are accountable to the data subjects for obtaining informed consent before collecting personal data.
- **Obligation 14: explicit consent to processing.** Controllers are accountable to the data subjects for obtaining explicit consent before collecting sensitive personal data.
- **Obligation 15: explicit consent to processing by joint controllers.** Controllers are accountable to the data subjects for obtaining explicit consent before allowing joint data controllers to process their sensitive personal data.
- **Obligation 16: informing DPAs.** Controllers are accountable to the data protection authorities to inform that they collect personal data.
- **Obligation 17: informing about the use of sub-processors.** Processors are accountable to the controllers for informing about the use of sub-providers to process personal data.
- **Obligation 18: security breach notification.** Controllers are accountable to data subjects for notifying them of security incidents that are related to their personal data.
- **Obligation 19: evidence of data processing.** Processors are accountable to the controllers for, upon request, providing evidence on their data processing practices**.**
- **Obligation 20: evidence of data deletion.** Processors are accountable to the controllers for, upon request, providing evidence on the correct and timely deletion of personal data.
- **Obligation 21: data location.** Data controllers are accountable to the data subjects for the location of the processing of their personal data.

9 Index of figures

10 Index of tables