Theorising Monitoring: Algebraic Models of Web Monitoring in Organisations

Kenneth Johnson[1], John V. Tucker[2], and Victoria Wang[3]

[1] School of Engineering, Computer and Mathematical Sciences,
Auckland University of Technology,
Private Bag 92006, Auckland, 1142, New Zealand
kenneth.johnson@aut.ac.nz
[2] Department of Computer Science, Swansea University,
Singleton Park, Swansea, SA2 8PP, United Kingdom
j.v.tucker@swansea.ac.uk
[3] Institute of Criminal Justice Studies, University of Portsmouth,
St George's Building, 141 High Street, Portsmouth, PO1 2HY, United Kingdom
victoria.wang@portsmouth.ac.uk

Abstract. Our lives are facilitated and mediated by software. Thanks to software, data on nearly everything can be generated, accessed and analysed for all sorts of reasons. Software technologies, combined with political and commercial ideas and practices, have led to a wide range of our activities being *monitored*, which is the source of concerns about surveillance and privacy. We pose the questions: *What is monitoring? Do diverse and disparate monitoring systems have anything in common? What role does monitoring play in contested issues of surveillance and privacy?* We are developing an abstract theory for studying monitoring that begins by capturing structures common to many different monitoring practices. The theory formalises the idea that monitoring is a process that observes the behaviour of people and objects in a context. Such entities and their behaviours can be represented by abstract data types and their observable attributes by logics. In this paper, we give a formal model of monitoring based on the idea that behaviour is modelled by streams of data, and apply the model to a social context: the monitoring of web usage by staff and members of an organisation.

Keywords: context, monitoring, records, interventions, surveillance, organisation, employee monitoring, web monitoring, abstract data types, algebraic specification, streams

1 Introduction

Our professional, economic, social and personal lives are facilitated and mediated by software running on computers and networks. Software exists to input, process and output data and, in particular, the software that drives computers and networks generate extensive data about their own operations. Given the diversity and ubiquity of software, computers and networks, data on nearly everything is being created – intentionally and unintentionally. Contemporary technologies allow a digital approximation of the lives of people and objects to be imagined, if not created in practice. Clearly, data is stored, accessed, analysed, and classified for all sorts of reasons, and is shared and used for all sorts of purposes. Combined with a wide spectrum of political and commercial needs and practices, software technologies have led to a wide range of our activities being *monitored*. An established example is the monitoring data collected by companies to improve their sales and customer service, through loyalty cards and recommender systems. Commercial monitoring designed to understand customer journeys and service personalisation is one among many sources of international concerns about surveillance and privacy [6, 7]. Such opportunities for monitoring have expanded enormously with the growth of mobile devices and smart products.

Despite the fact that monitoring phenomena are ubiquitous, and monitoring is arguably the principal source of data that drives the development of data science, the nature of monitoring has been neglected theoretically. Earlier, in [4], we posed the general questions:

What is the nature and purpose of monitoring?
Do the diverse and apparently disparate monitoring systems have anything in common?
What role does monitoring play in understanding surveillance and privacy?

In [4], we began to answer the first two questions by proposing an abstract approach to monitoring that can identify and explore structures common to different monitoring systems. By reflecting on some monitoring examples, we have isolated some fundamental conceptual components of monitoring systems. However, monitoring abounds in *many* domains of science, engineering, manufacturing, infrastructure, and environment on the one hand and commerce, healthcare, management, security, and social and financial services on the other. Thus, we are at the beginning of our programme of exploration and theory building. Our immediate aims are to develop theoretical ideas about monitoring systems and their applications, formulate precise general questions, and make comparisons and useful classifications of monitoring systems. Hopefully, our theorising will help the analysis of both technical and sociological issues to do with the third question asked above about monitoring.

The theory introduced in [4] formalises the idea that monitoring is a process that observes the behaviour of people and objects in a particular context. Monitoring involves choosing data to represent entities and behaviour, properties to observe by testing the data, and a form of storage to record the results. Monitoring is all about data. Thus, entities and their behaviours are modelled naturally by abstract data types and their observable attributes by logical languages. In this paper we focus on a particular model of monitoring in which the behaviour of people and objects in a context are modelled by a streams of data, i.e., sequences of data indexed by time. To illustrate and test the theory, we apply the model to a new social context, namely: the monitoring of web usage by staff and members of an organisation or company. This monitoring example is but one of hundreds to be found in the workplace, but it is easy to appreciate and reveals features that seem to be widely applicable.

The structure of the paper is as follows. In Section 2 we explain the basic concepts and principles of our general approach to monitoring: context, monitoring, storage and interventions. This introduces a conceptual framework for thinking about monitoring, one which can be formalised in a number of different ways. In Section 3 we give one such formal way: a general model of monitoring that takes behaviours to be modelled by streams of data.

Next, we turn to case studies. In Section 4 we discuss aspects of monitoring in organisations and companies. In Section 5 and Section 6 we use the general stream model in Section 3 to develop stream models of web monitoring in organisations. In Section 7 we consider storage as an abstract data type. In Section 8 we give examples of interventions. Thus, we will present our ideas about monitoring in three forms: as informal intuitions, formal definitions, and applications in a case study. Finally, in Section 9, we look back on the monitoring and intervention stack and we make some remarks on necessary further developments.

We assume that the reader is familiar with the basic algebraic concepts used to model data types: *signature*, *algebra*, *expansion*, *reduct*, *congruence*, *term*, *homomorphism*, *equational theory*, *first order theory*, etc. and, indeed, their relevant abstractions such as *institutions*. Whilst all are relevant only a few will appear in this short exposition of our theory, which we will develop using algebras. We have chosen to keep our algebraic techniques very simple to focus attention on monitoring and to present it as a new topic for theoretical investigation.

2 Concepts and Principles of Monitoring and Interventions

2.1 The Approach

Our conception of monitoring is based upon the following principle:

Principle. *Monitoring is confined to the collection, evaluation and recording of observational data about the behaviour of entities. The outputs of a monitoring system are simply records of observations.*

Thus, a key idea in our analysis is that it is *only* concerned with data. Thanks to this principle, our theory of monitoring is a theory of data and we can use the theory of abstract data types for its development.

The theory is based upon the idea that monitoring is a process that observes entities – people and objects, real and virtual – confined to a narrowly defined context wherein they and their behaviour can be represented by data that can be captured, queried and evaluated. A *context* consists of (i) *entities* and certain information about them called *characteristics*, and (ii) *behaviours*. Monitoring a context begins by choosing specific *attributes* of the data to be observed and a means of making *judgements* about the attributes.

To capture commonalities of diverse domains, and increase the generality of the analysis, we separate the acquisition of the monitoring data from its use. However, monitoring usually has a specific purpose. The records are inspected and certain properties trigger actions that may change the behaviour of the entities. We call these checks and changes *interventions*.

The components are illustrated in Figure 1. The monitoring system and the interventions are composed exclusively of data. We can design and combine abstract data types into what we call the *monitoring and intervention stack*.

Outside the monitoring and intervention stack we allow monitoring and intervention to employ any kind of technology and practice. To complete the description of monitoring, we introduce an informal notion of a

(i) *monitoring infrastructure*, for the technological and human systems that obtain and send the data representing the behaviour of entities to a monitoring system; and

(ii) an *intervention infrastructure* for the systems that receive the information from a monitoring system with interventions and initiate various responses and actions on the entities.

The details of these infrastructures are *not* part of the theory. Let us expand on the ideas introduced above.

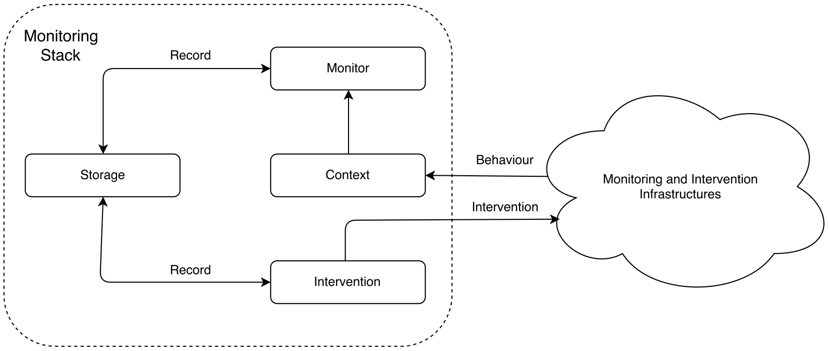

Fig. 1. The Monitoring and Intervention Stack Architecture

2.2 Conceptual Framework for Monitoring

The components that constitute the framework are these:

Context: Entities, Characteristics and Behaviour. Monitoring takes place in a context. A *context* is composed of *entities*. The entities have *characteristics* that define information that are relevant to the entities in the context. Entities have *behaviours* that can be observed. The behaviour of an entity depends upon the characteristics of the entity: characteristics are a parameter of entity behaviour.

Observation: Attributes and Judgements. Behaviours have *attributes* that can be observed. Observation involves making a query about, or testing, the behaviour of an entity for the presence of an attribute and making an evaluation. The evaluation produces a *judgement*.

In most cases the attribute will be a property that is assessed using a range of values on a scale; the evaluation will be a labelling, grading, measure, or probability. For example, physical measurements with error margins can give judgements that are bands of numerical values. Or a judgement may be a qualitative assessment based upon bands labelled metaphorically, such as the commonly used three-valued traffic light signifiers

$$\{green, amber, red\}.$$

In questionnaires and social surveys, five-valued assessments are used, such as:

$$\{strongly\ disagree, disagree, neutral, agree, strongly\ agree\}.$$

Monitoring and Records. The purpose of monitoring is to make an *observation* and a *record* of the observation. A record should contain the entity, its characteristics, the attributes observed and the judgements.

Interventions. To use the monitoring data, specific properties must be recognised in the records, noted and communicated to infrastructures *outside* the monitoring system, namely the intervention infrastructures. These communications with the outside we call *notifications*. The notification may initiate a series of physical or virtual actions that change an entity's characteristics and its behaviour.

Interventions are based on judgements. The judgements in a monitoring record are inspected: observations requiring actions are detected by *trigger conditions*. A trigger takes as input a judgement and returns a Boolean value that may lead to a notification for action. An *intervention* is a rule of the form

$$\textbf{if } \textit{trigger} \textbf{ then } \textit{action} \textbf{ else } \textit{do nothing}.$$

Triggers decide what should happen. Actions change the information in the entity's characteristics.

These ideas together form a general conceptual framework for monitoring that can be developed in a number of ways. Clearly, there are different semantic models of behaviour, different logics to formalise attributes and judgements, and different models of computation to analyse monitoring and interventions – see Section 9.

3 Monitoring Behaviour Modelled by Streams

We suppose the behaviour of entities takes place in time. We model the behaviour of an entity over time by a *stream* of data from A,

$$\ldots, a(t), \ldots \in A \text{ for } t \in T,$$

where T is a set of data that mark points in time. There are a number of forms of stream behaviour, as time and data can be discrete or continuous, and be structured by orderings and topologies. Here is the formal model, component by component.

3.1 Monitoring Streams

Time. The choice of T is influential and will be called the *monitoring clock*. We choose time to be discrete and take $T = \{0, 1, 2, \ldots\}$.

Behaviour. Behaviour is characterised by some data from a set A – typically quantitative measurements, text, images, audio or video. The streams are sequences in discrete time that are infinite and always well-defined:

$$a(0), a(1), a(2), \ldots, a(t), \ldots \in A \text{ for } t \in T.$$

Thus, we define a *stream* of data to be a *total function* $a : T \to A$ mapping time points in T to data in A. The space of all behaviours is the set $[T \to A]$ of all streams.

Contexts: Entities, Characteristics and Behaviour. Let E be the set of entities and let C be the set of characteristics. We define the behaviour of an entity as a stream of data generated by the behaviour map

$$[\![-,-]\!] : E \times C \to [T \to A] \tag{1}$$

such that for entity $e \in E$, with characteristics $\chi \in C$, at time $t \in T$,

$$[\![e, \chi]\!](t) = \text{data characterising behaviour of entity } e \text{ with characteristics } \chi \text{ at time } t.$$

Observation: Attributes and Judgements. Behaviours have attributes that can be observed over time. Let *Attr* be a set of attributes of behaviours. Let J be a set of judgements; often, J is a finite set.

Whilst observation may take many forms, the act of observing the behaviour of an entity, and making an evaluation, *starts* with a map

$$Obs_0 : Attr \times [T \to A] \to J. \tag{2}$$

We will need some variations:

Individual Observation. Suppose an attribute may vary according to the entity and its characteristics, and is specified by

$$P : E \times C \to \mathit{Attr}. \tag{3}$$

In this case, on substituting into mapping (2), the observation map becomes

$$\mathit{Obs} : E \times C \times [T \to A] \to J, \tag{4}$$

which, given an entity e with characteristics χ, computes the extent or degree that $P(e, \chi)$ is a property of $a \in [T \to A]$:

$$\mathit{Obs}(e, \chi, a) = \mathit{Obs}_0(P(e, \chi), a). \tag{5}$$

Observation over intervals and at instants. Since behaviour is time dependent, attributes are likely to involve time. First, it is common to look at behaviour over, say, an interval $[t_1, t_2] \subset T$. This suggests further adaptations of mapping (2) of the form

$$\mathit{Obs} : \mathit{Attr} \times [T \to A] \times T \times T \to J. \tag{6}$$

To observe the complete history of the entity up to time t we take the interval $[0, t]$. To observe behaviour at a particular time t we take the interval $[t, t]$. In both of these cases we adapt map (6) to one of this form

$$\mathit{Obs} : \mathit{Attr} \times [T \to A] \times T \to J. \tag{7}$$

Secondly, in time dependent cases, the attribute to be checked may depend upon the entity and its characteristics (as above), and upon the time of inspection.

Monitoring. Now, we implement monitoring as follows. First, let

$$R = E \times C \times \mathit{Attr} \times J \tag{8}$$

be the set of *records*.

Individual Monitoring. If the attribute P to be observed depends upon entity e and characteristic χ then $P(e, \chi)$ is the attribute to be checked. Using mappings (1) and (4), the monitor function becomes

$$Monitor : E \times C \to R \tag{9}$$

defined by

$$Monitor(e, \chi) = (e, \chi, P(e, \chi), \mathit{Obs}(e, \chi, [\![e, \chi]\!])).$$

Monitoring over intervals and at instants. If the attribute P to be observed depends upon entity e and characteristic χ then using mappings (1) and (6), the monitor function becomes

$$Monitor : E \times C \times T \times T \to R \tag{10}$$

defined by

$$Monitor(e, \chi, t_1, t_2) = (e, \chi, P(e, \chi), Obs(e, \chi, [\![e, \chi]\!], t_1, t_2)).$$

Monitoring Stream. Setting $t_1 = t_2 = t$, produces the mapping (7), which gives rise to a stream of records

$$monitor : E \times C \to [T \to R] \tag{11}$$

defined by

$$monitor(e, \chi)(t) = (e, \chi, P(e, \chi), Obs(P(e, \chi), [\![e, \chi]\!], t)).$$

3.2 Storage

The storage component is designed to collect and retain all, or some, of the records generated by monitoring. Thus, we assume that at any time the storage contains a finite list of records:

$$r_0, r_1, \ldots, r_n.$$

The index n is a function of time measured by the monitoring clock. For simplicity and brevity, we will assume that *all records generated by monitoring are stored*. The output of the monitoring component and the input of the storage component is a stream defined by map (11). It is easy to imagine cases where some filtering predicate chooses to store or not to store a record.

Clearly, there are many ways of designing a storage component, not least those of the theory of databases [5]. For our needs we may suppose there is an abstract data type for storage that comprises operations that input, organise, and output the records, and can support programs that can analyse the data in the store.

3.3 Interventions for Streams

Following the conceptual framework, interventions are based upon judgements, they do *not* involve behaviours directly, and so they are independent of the streams.

Triggers. Trigger conditions accept as input a judgement value $j \in J$, obtained as a result of the observations made by the function *Obs*, and outputs a truth value:

$$tc : J \to \mathbb{B}.$$

We denote the set of all trigger functions by $Trig = [J \to \mathbb{B}]$.

Actions. An action function

$$act : C \to C$$

performs an update $act(\chi)$ to the information $\chi \in C$. We denote the set of all action functions by Act.

Interventions. We use triggers and action functions to specify the intervention that results from the observation of an entity's behaviour. With both of these functions, we define an intervention of the form

$$tc \to act$$

where $(tc, act) \in Trig \times Act$. Mathematically, for $Intv = Trig \times Act$, we define the function

$$Int : R \times Intv \to E \times C \tag{12}$$

defined by

$$Int((e, \chi, P(e, \chi), j), tc \to act) = \begin{cases} (e, act(\chi)) & \text{if } tc(j) \\ (e, \chi) & \text{if } \neg tc(j). \end{cases}$$

4 Monitoring in Organisations

We now turn to organisational monitoring to illustrate the concepts of the theory.

4.1 Why Employee Monitoring?

Many people while at work are tempted to use the web for non-work purposes. Whilst most such activities are innocent and beneficial, some could be undesirable and even harmful to employees and to their organisations. For example, the use of pornographic and gambling sites would cause damage to the reputations of employees and organisations. Whether innocent or otherwise, employers are not pleased with the cultural effects of poor work ethics, or with financial losses associated with wasted time. Thus, employers have adopted surveillance technologies to monitor their employees' internet usage at work. The main aims for monitoring employee internet use during working hours are:

1. to deter unnecessary waste of contracted working hours on personal internet use;
2. to perform duty of care and protect employees from harmful exposure to the internet; and
3. to prevent any potential security breaches due to employee use of the internet.

(Compare [10].) Indeed, the use of monitoring across an increasingly wide range of behaviours is firmly established in organisational security management practices, and is commonly referred to as employee monitoring [18]. Of course, there are many issues associated with employee monitoring, especially legal and ethical. In the UK, as laid down by data protection laws, employers have the right to monitor employees' activities via different kinds of monitoring measures. These include opening mails and e-mails; using automated software to check e-mails; and checking logs of websites visited [2]. At the same time, the UK's data protection laws also set forth various rules that employers need to observe while monitoring, and the most important one is that – except in certain extreme cases – employers need to inform employees that they are monitored, what is being monitored, and why monitoring is undertaken (*ibid.*).

Of course, checking up on employees is nothing new [8]. Nevertheless, modern surveillance technologies present significant challenges because employee monitoring is now far more widespread, continuous, intense and secretive [3]. As a result, on the one hand, opponents of employee monitoring, such as labour unions, civil liberty groups, privacy advocates, and many employees themselves, have complained vehemently about employee monitoring [18]. The main charges include: increased levels of stress, decreased job satisfaction, decreased work life quality, lowered levels of customer service, creation of a hostile workplace, invasion of employee's privacy, lower morale, and unfair treatment of some employees (*ibid.*).

On the other hand, supporters of employee monitoring emphasise that employee monitoring not only increases productivity but also protects organisations from legal liability, data leakage and industrial espionage [12]. However, research on monitoring at the workplace also indicates that although a higher level of surveillance leads to more productivity on a task, it also leads to a lower level of quality of work on the task, and to a reduction of shared identity in the organisation's culture [9].

4.2 What Might Be Monitored and How?

Although the degree of employee monitoring is up to each employer, with advanced surveillance technologies, employee monitoring can be carried out very comprehensively. An employer can thoroughly monitor computer related activities of all of his/her employees down to the keystroke. Generally, the main aspects that are monitored include:

- applications;
- system settings;
- online surfing;
- connections to diverse devices and their locations;
- emails; and
- cloud activities.

The monitoring of these six aspects have objectives that are both facilitating and constraining. For applications, monitoring might be carried out to ensure appropriate versions of tools are in use, or to prevent the use of inappropriate tools, such as the Tor browser.[4] For system settings, monitoring might enhance access to peripherals (e.g., adding printers) or correct security vulnerabilities.

[4] https://www.torproject.org.

For online surfing, monitoring might concern website visits, specific page views, downloads, and audio and video streaming. For connections to devices, monitoring might extend standards and good practices of the office to a diverse range of mobile laptops, tablets and phones. For emails, monitoring is concerned with the identity of correspondents and the content of their messages. For cloud activities, monitoring is concerned with the services used.

All these six aspects can be investigated by analysing a wide range of log files that record the activities of the system, the network and the user. These aspects can also be investigated by desktop monitoring tools [13]. Desktop monitoring is a form of monitoring that targets a specific computer and every single action taken by its user. Surveillance software can be installed, directly into the targeted computer, or remotely, to intercept signals emitted by this target computer. It allows the monitoring of a target computer's usage, both online and offline. In general, the system administrator of an organisation is responsible for observing the data obtained via desktop surveillance, who might be asked to look for particular inappropriate actions.

4.3 Organisational Web Monitoring

How does a simple case of web monitoring in an organisation fit the conceptual framework outlined in Section 2? Although employee monitoring is the monitoring of people, it is actually the monitoring of data about people. To monitor their internet activity, the context is the behaviour of their computer accounts, and the devices they use to access them. Consider web monitoring as the real-time monitoring of an employee's online activity. A web monitoring system may have the following form:

- **Context:** Organisation/company;
- **Entities:** Employees via employee IT accounts;
- **Characteristics:** Account access permissions and status; monitoring status;
- **Behaviour:** Web activity;
- **Attributes:** Observation of sites visited classified as: *Unrestricted; Adult and Sexually Explicit; Gaming and Gambling; Personals and Dating; Proxies and Translators; and Intolerance and Hate*;
- **Record:** Staff account; data on websites visited, classified according to attribute categories;
- **Intervention:** In real time react to URL requests as follows: *allow; allow and notify employer; deny; deny and notify employer.*

Other categories that could be of interest to track are sites for Personal Email, Advertisements and Popups, Travel, P2P Downloads, or Social Networks. The categories need not have the same degree of interest for the organisation, e.g., shopping for a partner may be tolerated more than extremist politics.

5 Stream Model of Organisation Monitoring: Context

Using the general method given in Section 3, we construct algebraic models for monitoring web usage and data downloads.

5.1 Organisation: Entities, Identity and Characteristics

Entities. The *entities* we are monitoring are the computer accounts of people working at an organisation who require access to the internet.

Let *Empl* denote the set of employees. In many scenarios, each employee is assigned a personal account that uniquely identifies the employee and all the IT resources they access, many of which are communal. The employee's account stores or has links to a great deal of information about the employee, not all of which is available to the employee: for example, name and address; pay, insurance and tax details for employment; staff status and access permissions; professional development file; personnel file; and historical logs of recent activity.

Let *Acc* be the set of computing accounts. Define the injective function $h : Empl \to Acc$ that assigns a unique account $h(e) \in Acc$ to an employee $e \in Empl$ which allows access to the IT infrastructure. The gap between employee and account is important conceptually: in many case studies, *the objective is to monitor people but what is monitored is the behaviour of objects, namely devices and their use.*

Characteristics. In our case studies, we take the characteristics of an entity to be information about the employee's compliance with regulations. The first characteristic is *access status*, which refers to computing regulations that include or imply a classification of the content accessible via the internet in an *acceptable use policy* for their IT infrastructure. The second characteristic is the *monitoring status* specifying information about the current compliance of the employee's account.

Let τ denote the access status characteristic and μ the monitoring status. The characteristics of a monitored employee account is a pair

$$\chi = (\tau, \mu)$$

that will depend upon the account and what is monitored. Let C denote the set of all characteristics.

5.2 Web Examples

We will apply our monitoring stack to these monitoring scenarios.

Example 1: Monitoring Web Content. In this scenario, monitoring is used to determine employee compliance to the terms and conditions of their account. The access status is based on a specification that categorises web content, e.g., sites may be classified as in 4.3:

1. unrestricted,
2. adult/sexually explicit,
3. gaming and gambling,
4. personals and dating,

5. proxies and translators, and
6. intolerance and hate.

The access status defines numerical limits to the amount of web requests for content in each category. The employee's monitoring status records the number of times websites have been requested for each category. Over time, the status profiles the employee's (undesired) browsing habits.

Example 2: Monitoring Data Usage. The access status τ contains rules specifying data transfer limits applicable to downloadable content. The monitoring status μ is *green*, *amber* and *red*, respectively, corresponding to the situation where the employee has stayed within, or is close to, or has exceeded the data limit allowable for their account.

5.3 Modelling Web Behaviour

We will work through the components of the general stream model. The two monitoring contexts are completed by adding behaviour maps of the form

$$[\![-,-]\!] : Acc \times C \to [T \to A] \tag{13}$$

that deliver data from the monitoring infrastructure. The behaviour in time of account k with characteristics χ is a stream $[\![k,\chi]\!] = b$; what exactly is b? To model behaviour as a total function $b : T \to A$ we need to define the monitoring clock T and the data in the set A for the two examples.

Over time, the employee accesses the web, creating a stream of access requests. In both examples, time $T = \{0, 1, 2, \ldots, t, \ldots\}$ counts requests made by the account.

Web Content. To model web content as described in Example 1 in 5.2, we choose a finite alphabet W of symbols such that the set W^* of all finite words over W allows the formation of all possible uniform resource locators (URLs). We set $A = W^*$ and model account behaviour as a total function $b : T \to W^*$ such that $b(t) = w$ is the t^{th} webpage requested.[5]

Data Usage. Similarly, to model data usage in Example 2 in 5.2, we set $A = W^* \times \mathbb{N}$ and model behaviour as a total function $b : T \to W^* \times \mathbb{N}$ such that $b(t) = (w, n)$, where w is the web address requested and n represents the amount of data transferred as a result of the request (measured, say, in bytes).

Now, data is never without operations and our choice of data for A requires us to define a range of operations. Here is an operation to compute data usage over a specific time interval $[t_1, t_2]$, where $t_1, t_2 \in T$ and $t_1 < t_2$. Let $\pi : [T \to W^* \times \mathbb{N}] \to [T \to \mathbb{N}]$ be such that

[5] URLs are definable by a context free grammar.

$$\pi(b)(t) = \text{the data usage for behaviour } b \text{ at time } t.$$

Then, the aggregation operation $agg : [T \to W^* \times \mathbb{N}] \times T^2 \to \mathbb{N}$ is defined for $t_1 \leq t_2$ by

$$agg(b, t_1, t_2) = \sum_{t=t_1}^{t_2} \pi(b)(t). \tag{14}$$

6 Stream Model of Organisation Monitoring: Observation, Judgement, Monitoring

The next step is to specify the observation functions from which we derive the monitoring functions. Observation is based on a map of the form (2),

$$Obs_0 : Attr \times [T \to A] \to J,$$

and, since we have specified T and A already, to take this next step we must specify attributes Att and judgements J. We treat the two examples separately.

6.1 Web Content Observation

Attributes. Recalling Sections 5.2 and 5.3, here we are to observe, classify and record web pages. Classification is the key concept: let

$$W_1, \ldots, W_n \text{ where } W_i \subseteq W^*$$

be a collection of sets that classify URLs along the lines of Section 5.2. Let $W_0 = W - \bigcup_{i=1}^{n} W_i$ be the unrestricted websites.

Let $l_1, \ldots, l_n$ in $\mathbb{N}$ specify the maximum number of web requests for content allowed for each category, respectively; we write $\mathbf{w} = (W_1, \ldots, W_n)$ and $\mathbf{l} = (l_1, \ldots, l_n)$ in $\mathbb{N}^n$.

These features make up the access status characteristic $\tau = (\mathbf{w}, \mathbf{l})$. The outcome of tests make up the monitoring status μ.

The *attribute* used determines compliance with the limits $\mathbf{l}$ in τ and is defined by the test

$$P_{comply}(x_1, \ldots, x_n) \equiv (x_1 < l_1) \wedge \cdots \wedge (x_n < l_n).$$

For this scenario, we choose the set

$$J = \{\textbf{comply}, \textbf{non-comply}\}$$

of judgements and interpret them as

- **comply**, if the employee account has conformed to its limits,
- **non-comply**, if the employee account has overstepped its account limits.

The monitoring status μ is a value from J.

To test behaviour b for the attribute, we need some operations: define the characteristic function $\chi_i : W^* \to \{0, 1\} \subset \mathbb{N}$ by

$$\chi_i(w) = \begin{cases} 1 & \text{if } w \in W_i, \\ 0 & \text{if } w \notin W_i. \end{cases}$$

Next, define the map $f : W^* \to \mathbb{N}^n$ for $w \in W^*$ by

$$f(w) = (\chi_1(w), \ldots, \chi_n(w)).$$

The function logs the number of times a website appears in each category; note that a website may fall into none, one or many categories. For an unrestricted website $w \in W_0$, $f(w) = (0, \ldots, 0)$.

The pointwise lifting of f leads to the stream operation $F : [T \to W] \to [T \to \mathbb{N}^n]$ defined by the equation

$$F(b)(t) = f(b(t)). \tag{15}$$

Observation and Judgement. We determine if the observed behaviour on b satisfies the attribute P_{comply} between t_1 and t_2 using the function $profile : [T \to W^*] \times T \times T \to \mathbb{N}^n$ defined using vector addition on $\mathbb{N}^n$ by

$$profile(b, t_1, t_2) = \sum_{t=t_1}^{t_2} F(b)(t), \tag{16}$$

which applies the classification and counts web requests over the interval $[t_1, t_2]$. Computing $profile(b, t_1, t_2)$, we can define the observation function:

$$Obs(P_{comply}, b, t_1, t_2) = \begin{cases} \textbf{comply} & \text{if } P_{comply}(profile(b, t_1, t_2)) = \mathbf{t}, \\ \textbf{non-comply} & \text{if } P_{comply}(profile(b, t_1, t_2)) = \mathbf{f}. \end{cases}$$

As remarked earlier, if the categories are viewed with different degrees of concern then more complicated attributes can be defined that weight the categories W_i and apply different thresholds; recall Section 4.3.

6.2 Data Usage Observation

Attributes. Again, recalling Sections 5.2 and 5.3, here we are to observe, measure and record data downloads. In this scenario, both the employee's web requests and the amount of data are observed.

The access status characteristic contains a number $\tau \in \mathbb{N}$ representing the maximum amount of data that is allowed to be transferred over a period. The attributes are expressed by three tests on downloads:

$$\begin{aligned} P_{under}(x) &\equiv x < \tau - \epsilon \\ P_{near}(x) &\equiv \tau - \epsilon \leq x < \tau \\ P_{over}(x) &\equiv x \geq \tau \end{aligned}$$

where the constant $\epsilon \in \mathbb{N}$ defines an error margin around the account limit τ. Let

$$P_{usage} = (P_{under}, P_{near}, P_{over}).$$

We define the set of judgements $J = \{\mathbf{green}, \mathbf{amber}, \mathbf{red}\}$ to specify compliance to the data limit usage as follows:

- **green** under the limit,
- **amber** near to exceeding the limit, or
- **red** over the limit.

The monitoring status μ is a value from J.

Observation and Judgement. Next, we define the *Obs* operation to observe and judge the amount of data transferred via the account over time. Using the summation operation *agg* defined by equation (14), the data usage of the account over the period $[t_1, t_2]$ is computed by $agg(b, t_1, t_2)$. Thus, we define

$$Obs(P_{usage}, b, t_1, t_2) = \begin{cases} \mathbf{green} & \text{if } P_{under}(agg(b, t_1, t_2)), \\ \mathbf{amber} & \text{if } P_{near}(agg(b, t_1, t_2)), \\ \mathbf{red} & \text{if } P_{over}(agg(b, t_1, t_2)). \end{cases}$$

6.3 Records

The output of the monitoring component is a record of evaluating attributes over observed data. Mathematically, a record is an element

$$r = (k, \chi, P, j) \in R = Acc \times C \times Attr \times J$$

such that

- k is an employee account,
- χ is a characteristic of the employee account,
- P is a family of attributes,
- j is a judgement.

In our examples, records are created each time a web request is made, and a stream $\rho : T \to R$ of records may be obtained using map (11) in Section 3.1.

6.4 Monitoring

Applying a general construction of the monitoring operation (10) from Section 3.1, we have the monitoring map

$$Monitor : Acc \times C \times T \times T \to R$$

that is defined for entity k, characteristics χ and a family $P(e, \chi)$ of attributes over $[t_1, t_2]$ by

$$Monitor(k, \chi, t_1, t_2) = (k, \chi, P(k, \chi), Obs(k, \chi, [\![e, \chi]\!], t_1, t_2)).$$

From these we can derive streams of records.

Web content. For employee account k, characteristics $(\mathbf{w}, \mathbf{l}, \mu)$ and the family $P_{comply} = P_{comply}(k, \mathbf{w}, \mathbf{l}, \mu)$ of attributes:

$$Monitor(k, (\mathbf{w}, \mathbf{l}, \mu), t_1, t_2) =$$
$$(k, (\mathbf{w}, \mathbf{l}, \mu), P_{comply}, Obs(k, (\mathbf{w}, \mathbf{l}, \mu), [\![k, (\mathbf{w}, l, \mu)]\!], t_1, t_2)).$$

Data Usage. For employee account k, characteristics (τ, μ) and the family $P_{usage} = P_{usage}(k, \tau, \mu)$ of attributes.

$$Monitor(k, (\tau, \mu), t_1, t_2) = (k, (\tau, \mu), P_{usage}, Obs(k, (\tau, \mu), [\![k, (\tau, \mu)]\!], t_1, t_2)).$$

7 Stream Model of Organisation Monitoring: Storage

7.1 Histories, Thresholds and Queries

The records generated by monitoring all the accounts are collected and organised for retrieval by the storage component. Given the purpose of monitoring individuals, the organisation of the storage is based on accounts.

The storage is designed to preserve finite sequences

$$h = r_0, r_1, \ldots, r_s, \ldots, r_t$$

of records of the activity of an account $k \in Acc$, where r_s is the record received from the monitoring component at s within the monitoring period $[0, t]$. Each record in the list has k as its first component and such a sequence is called a monitoring *history* of the account k. The duration or length of the history h is $|h| = t+1$. The set $History_k$ of all possible histories of the account k is the basis of the abstract data type for storage.

Also associated with an employee account k are finite sequences

$$r_{t_1}, \ldots, r_s, \ldots, r_{t_2}$$

of records over an arbitrary monitoring interval $[t_1, t_2]$ called monitoring *transcripts* for the account k, where r_s is the record received at time s from the monitoring component, for $s \in [t_1, t_2]$. Let $Transcript_k$ be the set of all transcripts of the account k. Note $History_k \subset Transcript_k$.

In monitoring all the accounts in the organisation, we can define

$$History = \bigcup_{k \in Acc} History_k \text{ and } Transcript = \bigcup_{k \in Acc} Transcript_k.$$

The storage component specification needs to support high-level queries about the records it stores, and such queries can be considered part of the monitoring process. Thinking about programs that compute queries gives us a way of testing what data and operations may be needed when specifying an abstract data type for storage. The abstract data types can be analysed formally by applying models of computation designed for abstract data types [11, 14, 17].

For example, consider the simple queries that return

Q1 *A transcript of the employee's account in the last two hours;*

Q2 *A list of employee accounts exceeding the web content category threshold at least once; and*

Q3 *The percentage of employee accounts who have exceeded their download limit.*

We sketch the types of data and operations required to program answers to the queries above.

7.2 Data and Operations for Storage

Database Types. At the heart of storage is some form of database DB containing histories of all the accounts. A state of the database during monitoring can be modelled in different ways, such as by a map

$$\sigma : Acc \to History \text{ such that } \sigma(k) \in History_k, \text{for } k \in Acc.$$

In turn $History$, and $Transcript$, can be modelled as follows.

Let R_ϵ be the set R of records augmented by ϵ, a datum that denotes a *null record*. Let R_ϵ^* be the subset of the set $[T \to R_\epsilon]$ of all total maps, such that $h \in R_\epsilon^*$ has value ϵ almost always, i.e., $h : T \to R_\epsilon$ and $h(t) \neq \epsilon$ for only finitely many $t \in T$. Thus, formulating trivial conditions on the indexing, we have inclusions and embeddings:

$$History \subset Transcript \subset R_\epsilon^* \subset [T \to R_\epsilon].$$

At any moment, the number of accounts in the database σ is simply $|Acc|$.

Turning to the queries, Query Q1 involves three sorts: *employee accounts*, *transcripts*, and real-world *time*. Query Q2 uses *judgements* stored within records obtained by testing compliance with web content classifications. This query can return more than one matching employee account and so we make use of sets or lists Acc^* of accounts. The last query Q3 needs the *rational numbers* $\mathbb{Q}$ for quantitative operations; in this case, percentage.

In essence, we are sketching an abstract data type for the storage component based upon sets of data DB, Acc^*, $History$, $Transcript$, and $\mathbb{Q}$. Remember that R is comprised of four sets of data, Acc, C, $Attr$ and J, denoting employee accounts, characteristics, attributes and judgements, respectively. Other queries will create the need for other data types.

Database Operations. The basic operations on DB are about input and output. When a new record r is received from monitoring the account k, it is added to the database and the current history of account k is extended. We define $input : DB \times Acc \times R \to DB$ so that $input(\sigma, k, r)(k) \in History_k$ is updated with $input(\sigma, k, r)(k)(|\sigma(k)| + 1) = r$ where $|\sigma(k)|$ is the duration of the current history. The output operation $retrieve : DB \times Acc \to History$ is defined such that $retrieve(\sigma, k) = \sigma(k)$, the history associated with account k stored in database σ.

Transcripts are important in monitoring and can be extracted from histories by an operation $trans : DB \times Acc \times T \times T \to Transcript$ so that $trans(\sigma, k, t_1, t_2)$ is the sequence of records of account k stored for the period $[t_1, t_2]$.

Iteration on the Database. Many programs will involve searching the database for employee accounts matching a given property (e.g., to answer Query Q2). This requires a means of enumerating the accounts.

Operations on Clocks. Querying the database for observations occurring within a time frame is common; e.g., query Q1. Thus, it is necessary to have operations to support queries involving various measurements of real-world time. In our algebraic framework, we model clocks abstractly using the natural numbers $\mathbb{N}$ to count or mark events of interest. Specifically, the monitoring clock counts observations and indexes the histories and transcripts. To connect to real-world time requires operations that map between clocks. One method is to use a fast system clock to include a real-world time stamp in each record.

8 Interventions

In our framework, interventions formalise rules defined on records. They determine actions performed on characteristics, which may in turn impact on future observed behaviour. Here we give simple examples of interventions that can be applied to the models we developed in Sections 5, 6 and 7. In practical scenarios, the intervention infrastructure involves social-technical mechanisms, such as complex hierarchical human structures within organisations (managers, committees and tribunals) that determine appropriate actions according to rules, practices and expediencies.

Web Categorisation Intervention. Consider an intervention to restrict an account's access to a category W_i of web pages when its most recent record is judged to have exceed the specified account limit l_i. Suppose this reduces the account limit so that future non-compliant behaviours are detected earlier or blocked altogether.

For the following, suppose that the database in the storage component has been queried to obtain the most recent record $r = (k, \chi, P, j)$ of a particular employee account k. The characteristic is of the form $\chi = (\tau, \mu)$ where $\tau = (\mathbf{w}, \mathbf{l})$ is the pair of web categories and their browsing limits, respectively. An intervention $tc \to a$ is applied to record r, using the function defined by (12), such that

- trigger $tc : J \to \mathbb{B}$ on judgements is defined by $tc(\textbf{non-comply}) = \mathbf{t}$ and $tc(\textbf{comply}) = \mathbf{f}$,
- action $act : C \to C$ on characteristics is defined by $act(\mathbf{w}, \mathbf{l}, \mu) = (\mathbf{w}, \mathbf{l} - \mathbf{b}, \mu')$, where $\mathbf{b} \in \mathbb{N}^n$ depends upon μ, and the status is reset to μ'.

Subtraction of $\mathbf{b}$ from $\mathbf{l}$ reduces the limits of web categories accessible by the account; if $l_i = b_i$ then the new limit is 0 and the category W_i is banned. The μ' leads to a notification to intervention infrastucture

Data Usage Intervention. Consider an intervention that updates a download limit. The characteristic χ contains the access status $\tau \in \mathbb{N}$, a number specifying the maximum amount of data to be transferred.

To illustrate our conceptual framework's flexibility, we extend the mathematical description of interventions to perform actions based on a collection of judgements. To this end, let J^* denote the set of all finite sequences of judgements from the set $J = \{green, amber, red\}$ and call a subset of J^* a *judgement pattern*. In the data usage scenario, consistent long term over-usage of data by an account can be expressed as a judgement pattern with many instances of *amber* and *red*; for example, the pattern could be expressed by a regular expression,

$$\mathbf{over} = (amber|red)^n,$$

for a fixed number $n \in \mathbb{N}$.

The intervention $Int : DB \times Intv \to E \times C$, mapping (12), can be extended to operate over transcripts, such as a record sequence r^* of daily observations of account k. An intervention $tc \to a$ is applied to r^* using the definition

- trigger condition $tc : J^* \to \mathbb{B}$ is defined by setting $tc(\mathbf{over}) = \mathbf{t}$ and $tc(J^* \setminus \{\mathbf{over}) = \mathbf{f}$
- action $act : C \to C$ is defined for $b \in \mathbb{N}$ by $act(\tau, \mu) = (\tau + b, \mu')$.

This simply increases in the limit set by τ by b and resets the status to μ'. There are many scenarios for which this form of intervention is relevant. For example, a customer of an internet service provider, such as a phone company, will analyse their data usage noting frequent limit breaches. These breaches can result in costly penalties or data transfer at premium rates. The customer needs to change their subscription to a tariff with improved terms and conditions for data usage, thus increasing their limit.

9 Concluding Remarks

Using a general conceptual framework for analysing monitoring, we have given a general mathematical model of monitoring based on modelling behaviour as streams of data, and applied it to monitoring web activities of members of an organisation. This is the second of our papers that aims to develop a general theory of monitoring; the first [4] introduced the conceptual framework and a simpler stream model, and applied the stream model to monitoring offenders for criminal justice jurisdictions. We know of no other attempts to establish a general theory of monitoring.

To end, we return to the general approach sketched in Section 2 and consider next steps in developing a theory of monitoring.

9.1 The Monitoring Stack

The monitoring and intervention stack consists of context, monitoring, storage and intervention. The core concepts of our approach informally described in Section 2.1 can be summarised by four signatures:

signature **Context**

sorts *entity*, *characteristic*, *behaviour*

operations $[\![-,-]\!] : \mathit{entity} \times \mathit{characteristic} \to \mathit{behaviour}$

signature **Monitoring**

imports Context

sorts *attribute*, *judgement*, *record*

operations $\mathit{obs} : \mathit{attributes} \times \mathit{behaviour} \to \mathit{judgement}$
$\mathit{monitor} : \mathit{entity} \times \mathit{characteristics} \times \mathit{attribute} \to \mathit{record}$

signature **Storage**

imports Monitoring

sorts *database*, *history*

operations $\mathit{input} : \mathit{database} \times \mathit{entity} \times \mathit{record} \to \mathit{database}$
$\mathit{retrieve} : \mathit{database} \times \mathit{entity} \to \mathit{history}$

signature **Intervention**

imports Monitoring

sorts $Intervention$

operations $\mathit{int} : \mathit{record} \times \mathit{intervention} \to \mathit{entity} \times \mathit{characteristic}$

Each component can be specified as an abstract data type using signatures, equations and algebraic semantics. However, as the different organisational monitoring scenarios illustrate (not least the discussion of storage), there is a great deal more to these data types. To build the abstract data types of the monitoring stack, we need lots of operations and tests customised to applications, together with auxiliary types, operations and tests that act as a foundation for the construction; these latter can be grouped together into what we call the *platform* *algebra* for the stack. Some of these building blocks are generic such as algebras of infinite streams; others are made as the models develop.

9.2 Next Steps

Unsurprisingly at this early stage, the ideas in this paper require further analysis and application. Theoretically, there is much to be done on

(i) algebraic specifications of the monitoring stack to better understand the conceptual framework and stream models;
(ii) semantic models of behaviour, including formal models of time and space;
(iii) languages and logics for attribute specification and judgements;
(iv) storage models and their analytics;
(v) computability and complexity of monitoring.

In connection with (i), we have purposely simplified our presentation of the models, even avoiding signatures and equations, not to distract from the intuitions and ideas upon which the theory is based. In the matter of (ii), the notion of context seems to us simple and versatile, and capable of several formal approaches in which behaviour is analysed differently, e.g., by process algebras highlighting non-determinism and concurrency. Indeed, streams can be modelled in different ways. Time is an important component and leads to multiple clocks and streams. For (iii), the specification and evaluation of attributes in one form or another is core business for most formal design methods and will strengthen the theory and its relevance for thinking about tools for monitoring. In case (iv), the records managed by the storage component commonly meet the standard V^3 criteria for big data: volume, velocity and variety. There are many old and new technologies for effectively processing large quantities of data, including new technologies for graph databases [1]. In case (v), the computability and complexity of monitoring are important *a priori*, and will need to be developed when deploying the theory. The computational theory of data streams is rich [15, 16].

There is also much more to be done on case studies. Our explorations of monitoring in criminal justice and organisations have been rewarding, helping to shape, test and refine the theory. However, each case study has many more details and variations to model, and there are huge new areas to enter, such as monitoring in manufacturing, retail, healthcare, and computer system administration, including cloud computing. As the case studies grow, the potential of these ideas to influence practice, through methods and tools, can be judged.

Our interest in monitoring was sparked by investigations into some social problems of surveillance and privacy. Our theory of monitoring needs to be expanded to analyse and classify degrees of surveillance and privacy. This would involve integrating a theory of identity for the entities.

The digital world we have created means that recording lives – professional and private; intentionally or accidentally – has become a widespread technical and social phenomena. The technologies have turned monitoring from a tool for understanding a complex problem rationally with the help of data, into an administrative and commercial addiction and an end in itself. It is the world's appetite for monitoring that drives its desire for data, the bigger the better. We believe monitoring systems to be a topic with considerable potential for general theories, diverse applications, and socio-technical insights. At this stage we are content to develop the theory for the pleasure of thinking.

These ideas were first presented publicly by one of us (JVT) at Gregynog in an invited lecture at WADT16; we thank the organisers for the invitation and colleagues attending WADT 16 for their encouraging and helpful comments. This work was partially supported by the EPSRC project *Data Release - Trust, Identity, Privacy and Security* (EP/N028139/1 and EP/N027825/1).

References

1. Angles, R.: A comparison of current graph database models. In: Data Engineering Workshops (ICDEW), 2012 IEEE 28th International Conference on. pp. 171–177. IEEE (2012)
2. The National Association of Citizens Advice Bureaux: Monitoring at work. www.citizensadvice.org.uk/work/rights-at-work/basic-rights-and-contracts/monitoring-at-work (2016)
3. Ford, M.: Surveillance and privacy at work. www.ier.org.uk/publications/surveillance-and-privacy-work (1998)
4. Johnson, K., Tucker, J.V., Wang, V.: Monitoring and intervention: Concepts and formal models. arXiv:1701.07484 (2016), https://arxiv.org/abs/1701.07484
5. Korth, H.F., Sudarshan, S., Silberschatz, A.: Database System Concepts. McGraw-Hill (2010)
6. Lyon, D.: Surveillance as Social Sorting: Privacy, Risk and Digital Discrimination. Routledge (2003)
7. Lyon, D.: Surveillance Studies: An Overview. Polity Press (2007)
8. Mishra, J., Crampton, S.: Employee monitoring: privacy in the workplace? SAM Advanced Management Journal 63(3), 4–11 (1998)
9. O'Donnell, A.T., Ryan, M.K., Jetten, J.: The hidden costs of surveillance for performance and helping behaviour. Group Processes and Intergroup Relations 16, 246–256 (2012)
10. Paganini, P.: Employee monitoring, a controversial topic. securityaffairs.co/wordpress/46814/digital-id/employee-monitoring.html (2016)
11. Stoltenberg-Hansen, V., Tucker, J.V.: Effective algebras. In: Abramsky, S., Gabbay, D., Maibaum, T. (eds.) Handbook of Logic in Computer Science. Volume IV: Semantic Modelling, pp. 357–526. Oxford University Press (1995)
12. Tavani, H.: Ethics and Technology: Controversies, Questions, and Strategies for Ethical Computing. John Wiley & Sons (2010)
13. TechTarget: Comprehensive guide to desktop monitoring tools. searchenterprisedesktop.techtarget.com/essentialguide/Comprehensive-guide-to-desktop-monitoring-tools
14. Tucker, J.V., Zucker, J.I.: Computable functions and semicomputable sets on many sorted algebras. In: Abramsky, S., Gabbay, D., Maibaum, T. (eds.) Handbook of Logic in Computer Science. Volume V: Logical and Algebraic Methods. Oxford University Press (2000)

15. Tucker, J.V., Zucker, J.I.: Continuity of operators on continuous and discrete time streams. Theoretical Computer Science 412, 3378–3403 (2011)
16. Tucker, J.V., Zucker, J.I.: Computability of operators on continuous and discrete time streams. Computability 3, 9–44 (2014)
17. Tucker, J.V., Zucker, J.I.: Generalizing computability to abstract algebra. In: Sommaruga, G., Strahm, T. (eds.) Turing's Revolution: The Impact of His Ideas About Computability. Birkhäuser, Springer Basel (2015)
18. Yerby, J.: Legal and ethical issues of employee monitoring. Online Journal of Applied Knowledge Management 1(2), 44–55 (2013)