Semantics and Provenance for Accountable Smart City Applications

Heather S. Packer, Dimitris Diochnos, Michael Rovatsos, Ya’akov Gal, and Luc Moreau

Abstract. The recent media focus on Smart City services, particularly ride sharing, that provide ordinary users with the ability to advertise their resources has highlighted society’s need for transparent and accountable systems. Current systems offer little transparency behind their processes that claim to provide accountability to and for their users. To address such a concern, some applications provide a static, textual description of the automated algorithms used, with a view to promote transparency. However, this is not sufficient to inform users exactly how information is derived. These descriptions can be enhanced by explaining the actual execution of the algorithm, the data it operated on, and the parameters it was configured with. Such descriptions about a system’s execution and its information flow can be expressed using PROV, a standardised provenance data model. However, given its generic and domain-agnostic nature, PROV only provides limited information about the relationship between provenance elements. Combined with semantic information, a PROV instance becomes a rich resource, which can be exploited to provide users with understandable accounts of automated processes, thereby promoting transparency and accountability. Thus, this paper contributes, a vocabulary for Smart City resource sharing applications, an architecture for accountable systems, and a set of use cases that demonstrate and quantify how the semantics enrich an account in a ride share scenario.

Keywords: Provenance, Semantic, Accountability, Transparency, Ride Share

1. Introduction

A Smart City is an emerging conceptual view of a city that promotes the use of information and communication technologies to engage with citizens to develop social capital and intellectual capital, to make better use of hard infrastructure, to reduce usage of environmental capital, and to support smart growth. In this context, a class of promising online applications seek to enable citizens to share services in smart cities. This class of applications, which is referred to as smart sharing, are varied and include sharing care sharing (Uber, Blablacar, Lyft), bus routes (Chariot), parking space (JustPark), spare rooms (airbnb), and cleaning services (HomeJoy). Smart sharing services allow citizens to benefit from customised and financially advantageous offerings, potentially reducing environmental impact.

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

0000-0000/09/$00.00 © 2009 – IOS Press and the authors. All rights reserved
Smart sharing services are not just online platforms since they mediate access to real people and physical resources. It is recognized that smart sharing services have inherent entry barriers, because they “rely on customers and hosts overcoming their fear of strangers.” Thus, smart sharing services attempt to delay their users’ fears through a variety of techniques, including screening processes, security measures, reputation systems, and incentive mechanisms. For instance, due to heavy media attention, some ride share applications have been banned in various locations around the world because they “do not enough to protect their passengers from unlicensed drivers.” In response, they further provided driver screening, insurance, feedback and reputation systems.

Two strong characteristics are now commonly supported by smart share services: transparency and user accountability. Transparency is operating in such a way that when a shared resource is offered to a user, it is required that all details of the resource are exposed to its potential consumers so that they can make informed decision as to whether to use the resource or not. By making users accountable for their actions (or the quality of their resources), it is believed that users are less likely to make mistakes or take actions that may affect the quality of the service. User accountability is usually a by product of a reputation system and transparency, which act as an incentive mechanism for users to provide good quality services.

Generally, companies offering smart sharing services rely on automated processes, from matching users to services, to “algorithms that identify suspicious behavior.” Automated processes can seem like a black box to users. Typically, these service-oriented systems rely on feedback from users, which may be viewed by other users or to compute reputation recommendations. Some applications may provide a static description of the algorithm used to compute reputation in order to be transparent. However, this is not sufficient to inform users exactly how a reputation measure is derived. Thus, in other words, there is still a lack of accountability and transparency about the processes used by smart share applications: have due processes been followed, has screening been applied, how are resources matched to consumer’s preferences, how is reputation computed, and how does feedback affect reputation? These issues have been reported in a number of news articles.

It is important that not just users are held accountable for their action but the systems are too. There are two reasons for this: (1) transparent and accountable systems increase a user’s understanding of its processes, and therefore can increase the user’s trust of the system; and (2) accessible accounts enable user’s to feel that their actions can be monitoring, thus incentivising them to conduct themselves in a respectable manner. Weitzner et al. advocate new governance and frameworks to hold people accountable for the misuse of data, and suggest that provenance can help towards this aim. Specifically, provenance is a record that describes the people, institutions, entities, and activities involved in producing, influencing, or delivering a piece of data or a thing. PROV is a W3C standard for provenance, which provides a domain-agnostic information about the relationship between provenance elements, which can be a PROV entity, activity or agent. However, combined with semantics, PROV becomes a rich resource which can be exploited to provide users understandable accounts of automated processes, promoting transparency and accountability of smart sharing services themselves.

Provenance data can be hard for both expert and non-expert users to understand because of its technical content, and its potential scale and complexity. However, PROV provenance, given its well-defined nature, is well suited to construct narratives. The role of narratives is to provide an account of connected events, which can be organised in a number of different categories. Textual narratives can be used to improve

---

2. URL: [http://www.bbc.co.uk/news/magazine-21339891.html?_r=2&abt=0002&abg=1](http://www.bbc.co.uk/news/magazine-21339891.html?_r=2&abt=0002&abg=1)
3. URL: [http://blog.uber.com/uberpopfactsbxl](http://blog.uber.com/uberpopfactsbxl)
4. URL: [http://blog.uber.com/uberpopfactsbxl](http://blog.uber.com/uberpopfactsbxl)
5. URL: [http://the guardian - Uber taxi service banned in Berlin on safety grounds](http://www.theguardian.com/technology/2014/aug/14/uber-taxi-service-banned-berlin-safety-grounds)
6. URL: [https://www.lyft.com/safety](https://www.lyft.com/safety)
transparency because they explain who performed processes and their inputs and outputs.

This paper posits that using semantics to enrich PROV D-M [20] to automatically generate sentences using templates, improves PROV’s supports both systems transparency and accountability. It presents a framework for accountability, which is comprised of services, one of which is a provenance enabled reputation service, and an explanation service which provides a narrative of a provenance subject. This work is the first to use provenance, semantics, and narratives, to provide an account of how documents, data, and information are used, modified, and created by both users and services. Extant state-of-the-art frameworks tend to focus on logging this information, but do not present it in an easy to understand format to users. Presenting information in an accessible format allows all types of users to audit and become aware of their actions within a system. This paper argues that the role of provenance and its semantic markup is critical in building accountable Smart City applications. Specifically, this paper contributes towards the state-of-the-art:

1. A vocabulary specifically designed to support the markup of provenance to support accountability. It describes services, activities, and entities within a Smart City application.
2. A framework that supports accountability as a service, and includes the following components: (1) The reputation service, a PROV-enabled reputation system that uses the Smart City vocabulary to markup its provenance; (2) The explanation service, which provides a narrative on the provenance of a give piece of information using the terms defined in the Smart City application vocabulary.
3. Provenance explanation examples that demonstrate the benefits of using semantics to provide an account of provenance.

The remainder of this paper is organised as follows. Section 2 describes the background literature on accountable frameworks for distributed services and narrative for accountability. Section 3 then presents a vocabulary for Smart City applications. Following that, Section 4 describes an accountable framework and how it uses the vocabulary. Then, Section 5 presents a ride sharing scenario using the framework and vocabularies. Succeeding that, Section 6 details examples from the ride sharing scenario and narrative examples. Finally Section 7 provides conclusions.
under applicable rules.” They identify considerations for legal rules for accountable systems, including “(1) what degree of transparency rights should those subject to data mining have, (2) what will be the mechanism for the correction of data found to be incorrect, and (3) will there be legal recourse in the event agencies rely on incorrect information after the error has been pointed out by the subject”. This work highlights the need for users to be able to understand which information held is about them and how it is used.

Provenance data can be used for accountability [11, 19], provided that users can rely on the provenance record and authenticate its sources so that it can be used to assign credit or blame. Halpin [12] argues that provenance information can be used as the foundation for a model of privacy and trust in the context of the Semantic Web. Ruth et al. [24] discuss the use of provenance in a layered architecture for accountability in cloud computing. However, because of the limits of cloud computing, provenance techniques such as audit trails were not possible because it was not feasible to store all the versions of the resources referenced in the provenance records.

Existing work on building accountability into electronic commerce protocols bears some similarity accountability in Smart City applications. For example, Crispo et al. and Kailar et al. [7,15] describe frameworks for analysing accountability in communication protocols; [6] discusses the delegation of trust with full accountability for electronic commerce applications. The models presented in this work influence the vocabulary for accountable Smart City applications. Similarly, [23] introduces social computations, which aligns with the social component of Smart City applications. They introduce an abstract model for social computation, and relevant terms. These models and terms influence the proposed Smart City vocabulary.

Narratives are potentially the most accessible way to communicate information, they provide an account of connected events, which can be organised in a number of different categories. Research has shown that it is how users make sense of their own information [4,16,18], and it is effective to communicate with communities and individuals [8,17]. It is well suited to describing provenance data because of the well defined PROV.

Previously, Semantic Web technologies have been used to generate narratives [25,14,10]. In more detail, Tuffield et al. [25] and Jewell et al. [14] describe the OntoMedia ontology, which supports the generation of narratives. The work presented in [25] discuss approaches to generate narratives from a vocabulary, the approaches included are based on character, plot and user modelling. While, the work presented in [14] describes how OntoMedia is used to annotate the vast collection of heterogeneous media. The work [10] use ontological domain knowledge to select and organise a narrative discourse on an interest topic to a user.

3. Smart City Vocabulary

A vocabulary to describe Smart City actors, activities and entities is required to provide types to provenance elements, so that the types can be exploited by other services. The type information allows services to apply their own constraints, facilitating the leverage the information in the provenance.

The class of Smart Share applications envisaged for the Smart City involve agents, humans or otherwise, having capabilities they wish to offer as a service to other agents, typically humans those having specific requirements, in terms of baby-sitters, car space, or rides. For instance, the agents have been abstract using the notions of peers. Resolving a Smart Share problem involves the use of an orchestrator responsible for matching tasks to peers and scheduling their execution, resulting in the desired allocations of baby-sitters to families, or assignments of car spaces to drivers. To become accountable, a Smart City application needs to be able to describes how this allocation of tasks and their scheduling are achieved, and hence, it requires a vocabulary that involves notions of peers, capabilities, tasks.

Services shared via Smart City applications rely on reputation to set them apart from on another. Reputation is created by word of mouth or feedback from consumers of the service, and can be viewed by the community of a Smart City application. Reputation can also be provided by a set of consumers, a collective which represents a groups’ view on a service. To support services in managing feedback and reputation the vocabulary for Smart City provides notions of reputation reports, feedback reports, collectives, and a reputation peer.

Concretely, the Smart City vocabulary focuses on describing three components: (1) agents within a Smart City application, including users, peers, and collectives and their properties; (2) activities; and, (3) entities describing plans, tasks and messages. The namespace used for the vocabulary is http://
smartsociety-project.github.io/cas/ and it defines the following terms:

- **An agent** is anything that can perform an activity; alternatively, anything that has capabilities.
- **A user** is a person who is using a SmartSociety system.
- **A peer** is a software agent in a SmartSociety system that represents another agent.
- Agents, users, peers all have **identities**; an unauthenticated user gets an assigned identity.
- **A collective** is an agent that consists of multiple member agents.
- **An activity** is the condition in which things are happening or being done.
- **A capability** is a prospective, though not necessarily planned or agreed, activity.
- **A task** is something that involves capabilities, potentially contributed by several agents.
- **A plan** is a specification for the execution of a task.
- **A protocol** is a collection of plans that involve communications between peers.
- **A message** is a piece of information exchanged between agents.
- **A messaging action** is the constituent of a protocol: it involves information exchange and subsequent action.
- **A feedback report** is a report about a subject that contains a set of value rating categories. For example, "star rating = 4.3".
- **A reputation report** is a report about a subject that contains a set of value rating categories. For example, a subject has been rated with a set of star ratings, and the average of those star ratings are "average star rating = 4.3".
- **A reputation peer** is a peer that specialises in storing feedback about subjects and generating reputation about subjects.

4. Accountability as a Service

To make an application accountable, several key pieces of functionality need to be available: tracking application’s actions and decisions; providing explanations for all aspects of the system; managing feedback about all its components; and calculating their reputation. Such accountability functionality is non-trivial and requires substantial design and implementation efforts. It is therefore critical to make it reusable, so that it can easily be deployed in multiple Smart City applications.

Therefore, embracing service-oriented architectures [9], the concept of **accountability as a service** is proposed, which consists of exposing accountability-related functionality into a set of well-defined and reusable APIs (Application Programming Interfaces). Such APIs are intended to be application and domain-independent, and they can be deployed in Smart City application. These APIs can be held peers accountable for their actions; thus, this increases user’s trust and adoption of the system, and it also increase user’s awareness that their actions matter.

The role of the accountability service is to provide accounts of actions performed by Smart City application’s peers who are identified, and how entities are created, modified and viewed. It is important that order to support this, the framework has the following requirements:

1. Provide a record of peers, entities, and activities involved in Smart City applications, in particularly pertaining to the generation of automated processes such as reputation generation and matching users to services;
2. Provide a reputation service that manages feedback and reputation;
3. Provide accessible accounts to users to explain automated processes.

The framework to support accountability as a service is designed to support the above requirements, and it is comprised of the components (see Figure 1):

1. **ProvStore** which is a specialised service for storing provenance. It supports the first requirement by storing the provenance records detail the peers, entities and activities in Smart City Applications.
2. A set of peers which submits provenance to ProvStore detailing who performed which activities and where entities are derived from. This component also supports the first requirement, by providing provenance.
3. The reputation peer which stores feedback reports on subjects and computes the reputation reports based on feedback reports. It also submits provenance to ProvStore. This component supports the second requirement by providing a service to manage feedback and reputation.
4. The explanation peer uses provenance from ProvStore to generate a textual explanation of a
provenance element. This component supports the third requirement by generating narratives about a provenance subject.

The provenance submitted to ProvStore is described using PROV, which uses the PROV-DM conceptual data model. The type of a provenance element is defined using `prov:type` whose value is from the vocabulary defined in Section 3. These types are used by the explanation peer to provide more detailed information in the textual description of a provenance element.

The section is organised as follows, in Sections 4.1, 4.2 and 4.3 introduce ProvStore, Reputation Peer and the Explanation Peer, respectively.

4.1. ProvStore

ProvStore is a specialised service for storing PROV. It also enables users to query provenance and generates visualisations allowing users to analyse provenance data. ProvStore’s API\(^{10}\) provides a RESTful web service for the storage and access of provenance documents in various representations. Via the API, any client can manipulate documents contained in the store and explore and retrieve information from them. ProvStore stores documents containing provenance descriptions. A document can contain bundles which may be added via the API. A PROV bundle is a mechanism by which provenance of provenance can be expressed.

4.2. The Reputation Peer

The reputation peer provides Smart City applications, with a service that manages a subject’s reputation. It stores feedback reports, computes and stores reputation reports. Once the reputation peer receives a feedback report it generates reputation reports. Feedback reports are comprised of key-value pairs describing attributes about a subject, and meta-information about feedback which includes who or what it is about, the authors, and associated events (see Figure 2 for an example).

\[
\begin{array}{l}
\text{"feedback_id": 325,} \\
\text{"application_id": 1,} \\
\text{"event_id": 24,} \\
\text{"subjects": {"subject_1": 4},} \\
\text{"authors": {"author_1": 1},} \\
\text{"feedback": {} \\
\text\{"category_1": 5,} \\
\text\{"category_2": 5,} \\
\text\{"..\} \\
\text\{"category_n": 5} \\
\end{array}
\]

Fig. 2. An Example of a Feedback Report

The reputation report about a subject is derived from all the feedback reports posted to the peer. The categories in which a subject is rated is dependant on the categories are defined in the feedback value about that subject. The reputation peer calculates the average value of a category given a set of feedback reports (see Figure 3).

\[
\begin{array}{l}
\text{"reputation_id": 135,} \\
\text{"application_id": 1,} \\
\text{"subject": 4,} \\
\text{"feedback": {} \\
\text\{"average_category_1": 5,} \\
\text\{"average_category_2": 5,} \\
\text\{"..\} \\
\text\{"average_category_n": 5} \\
\end{array}
\]

Fig. 3. An Example of a Reputation Report

The provenance submitted by this peer will support the accountability of:

1. Which feedback report was related to which event and who;

\(^{10}\)ProvStore’s REST API: https://provenance.ecs.soton.ac.uk/store/help/api/
2. Which feedback reports were used to generate reputation and opinion reports;
3. Who accessed feedback, reputation, and opinion reports;
4. The security and privacy policies used to protect user’s information.

The reputation peer’s resources are exposed via a REST API. Specifically, the API allows the submission of feedback reports about a subject or collective of subjects, from an author or collective of authors.

4.3. The Explanation Peer

This peer provides a service that provides a narrative about a single provenance element (either an entity, activity or agent) in either a provenance document held by ProvStore or a set of provenance documents related to a specific application.

It uses sentence templates to explain the provenance data about a subject. These sentence templates use both the types defined in: the PROV D-M [21]; and, the smart cities vocabulary for provenance (see Section 3). If the provenance data is not marked up with the smart cities vocabulary, the peer by default reverts back to using sentence templates based on the PROV’s types, which are described in Table 1.

The templates P4-P13 in Table 1 use the relationship specified in the second column to identify the bindings for the sentence template variables. For example, if a subject is a prov:activity and has an association with a agent then the template p5 is used. It exploits the PROV’s Association relationship and uses the activity as the subject and the agent defined in this relationship to as variable binding to the P5 template, where wasAssociatedWith(subject, peer-1) is used to create the sentence “It was associated with agent/s peer-1” where “It” refers to the subject. Otherwise, it overrides these default sentences using the templates in Table 2. These templates differ from the those presented in Table 1 because they can use the semantics in the sentence template to define a path of relationships to connect two provenance elements. For example, the explanation peer given the subject feedback-1 uses the feedback report template in Table 2. First, it identifies the Smart City type as a feedback report using the provenance statement (feedback-1, [prov:type=‘provsm:feedback report’]). Second, it identifies generation relationships referring to that entity, where wasGeneratedBy(feedback-1, feedback_submission-1, -). Third, it then identifies which agent was associatedWith the activity, where wasAssociatedWith(feedback_submission-1, peer). This chain of relationships is used to identify the bindings for the feedback report’s sentence template, for example the template The {subject} is a feedback report, the feedback was left by the [peer/user/collective]. It was submitted using the {activity}, has the following bindings subject = feedback-1, [peer/user/collective] =peer, and {activity} = feedback_submission-1. If the provenance does not contain the required fields than it reverts back to using the template in Table 1. The Table 2 does not make the provenance paths used to bind the variables explicit because of conciseness, and instead uses the possible variable’s types for the bindings.

Specifically, in order to generate a narrative the explanation peer:

1. Identifies the template for the subject by using the type defined either in the PROV or by a Smart City type. If it uses the PROV type then it:
   (a) Identifies all the provenance elements that are connected to it by provenance relations.
   (b) Identifies the templates for the provenance elements that connect to the subject using their PROV type.

Or if it uses a Smart City type then:
   (a) Identifies all the provenance elements that are connected to the provenance subject described in the smart cities sentence template about the subject.
   (b) Identifies the templates from the Smart City templates to described the connect provenance elements.

2. The sentences about the subject and connected provenance elements are strung together to form a paragraph.

4.4. APIs Outline

The frameworks adopts a REST approach [1] for accountability as a service. Tables 3 summarises the key resources related to provenance, reputation, and explanations, as well as the REST Method allowed on them.

5. Ride Share

Ride Share is an application that enables car sharing for workplace workers, university students and simi-
lar large community members living in and around the Smart City. For example, at Ben-Gurion University, there are thousands of students coming from all over the country which has created a vivid car sharing community based on actual and acute travel needs. The application allows both drivers and commuters to offer and request rides. These offers and ride requests include details about required travels, timing, locations, capacity, prices, and other details relevant for car sharing. It performs automatic matching of commuters to available cars, by considering origin and destination, routes, capacity and other available information. Incentives are used to influence participant behaviours and maximise the global system goals.

Ride Share has been chosen because it uses automated algorithms for matching users in rides and generating reputation reports. The execution of these automated algorithms is documented using provenance, which is marked up with the smart cities vocabulary; provenance is exposed to users via the explanation peer. Without recorded provenance, smart cities vocabulary, and explanation peer, these automated process would be a black box to users.

The architecture of Ride Share is based on five core components (see Figure 4): a view, ride matching peer, reputation peer, and ProvStore, organised according to the layered architecture of Figure 1. The view provides the user with the graphical components with which to enter their ride requests, and to view and select potential rides. The matching peer provides matches containing drivers and commuters, which the users can select. The reputation peer and ProvStore are described in Section 4.

Table 1

<table>
<thead>
<tr>
<th>Number</th>
<th>Prov Types</th>
<th>Sentence template</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>Entity</td>
<td>The [subject] is a [provtype]</td>
</tr>
<tr>
<td>P2</td>
<td>Agent</td>
<td>The [subject] is a [provtype]</td>
</tr>
<tr>
<td>P3</td>
<td>Activity</td>
<td>The [subject] is a [provtype]</td>
</tr>
<tr>
<td>P4</td>
<td>Alternate</td>
<td>It was an alternate of [alternate/s]</td>
</tr>
<tr>
<td>P5</td>
<td>Association</td>
<td>It was associated with agent/s [agent/s]</td>
</tr>
<tr>
<td>P6</td>
<td>Attribution</td>
<td>It was attributed to agent/s [agent/s]</td>
</tr>
<tr>
<td>P7</td>
<td>Collections</td>
<td>It was a member of the [collection/s] collection/s</td>
</tr>
<tr>
<td>P8</td>
<td>Communication</td>
<td>It was informed by [agent/s]</td>
</tr>
<tr>
<td>P9</td>
<td>Delegation</td>
<td>It acted on behalf of [agent/s]</td>
</tr>
<tr>
<td>P10</td>
<td>Derivation</td>
<td>The [subject] was derived from the entity/ies [entity/ies]</td>
</tr>
<tr>
<td>P11</td>
<td>Specialisation and Revision</td>
<td>The [subject] was a specialisation of the entity (entity), and is a revision of [entity/ies]</td>
</tr>
<tr>
<td>P12</td>
<td>Generation</td>
<td>It was generated by the activity [activity/ies]</td>
</tr>
<tr>
<td>P13</td>
<td>Usage</td>
<td>It was used by the activity/ies [activity/ies]</td>
</tr>
</tbody>
</table>

Fig. 4. Layered Model of Ride Share

5.1. Feedback and Reputation Reports

Feedback reports can be submitted by both riders and drivers after a ride has been completed. Table 4 describes the feedback categories supported by Ride Share, in feedback reports.

A reputation report is a rating of a user structured according to reputation categories; such reputation categories are defined from the categories comprised in feedback reports submitted about the user. Each reputation category value is calculated by a simple function, averaging the values associated with the corresponding feedback category in feedback reports about the user (see Table 5). At this stage, the focus of this
### Table 2

<table>
<thead>
<tr>
<th>Smart City Type</th>
<th>Prov Type</th>
<th>Sentence template</th>
</tr>
</thead>
<tbody>
<tr>
<td>agent</td>
<td>prov:Agent</td>
<td>The {subject} is an agent within a Smart City framework, that has the capabilities {capabilities}. It performed the following activities {activities}.</td>
</tr>
<tr>
<td>user</td>
<td>prov:Agent</td>
<td>The {subject} is a human user that uses the Smart City framework. It has the capabilities {capabilities}. It is identified by {identities}, which is maintained by the framework. It is the member of the collective/s {collectives}. It performed the following activities {activities}.</td>
</tr>
<tr>
<td>peer</td>
<td>prov:Agent</td>
<td>The {subject} is a software agent in the Smart City framework. It has the capabilities {capabilities}. It is identified by {identities}, which is maintained by the framework. It is the member of the collective/s {collectives}. It performed the following activities {activities}.</td>
</tr>
<tr>
<td>identities</td>
<td>prov:Agent</td>
<td>The {subject} is the identify for the {agent/user/peer} and is maintained by the Smart City framework. It was used to identify the {agent/user/peer} in the following activities {activities}.</td>
</tr>
<tr>
<td>collective</td>
<td>prov:Agent</td>
<td>The {subject} is a collective, who has the following members {agents/users/peers}. It is an agent with the Smart City framework, that has the capabilities {capabilities}. It performed the following activities {activities}.</td>
</tr>
<tr>
<td>activity</td>
<td>prov:Activity</td>
<td>The {subject} is an activity performed by the {agent/user/peer/collective}.</td>
</tr>
<tr>
<td>capability</td>
<td>prov:Entity</td>
<td>The {subject} is a capability, specifically in this context it describes the capability of the {agent/user/peer/collective}.</td>
</tr>
<tr>
<td>task</td>
<td>prov:Entity</td>
<td>The {subject} is a task, it describes the capabilities {capabilities} possibly involving the following agents {agents/users/peers/collectives}.</td>
</tr>
<tr>
<td>plan</td>
<td>prov:Entity</td>
<td>The {subject} is a plan, it describes the a task that has the capabilities {capabilities} which involves the following agents {agents/users/peers/collectives}.</td>
</tr>
<tr>
<td>protocol</td>
<td>prov:Entity</td>
<td>The {subject} is a protocol that describes the a task that has the capabilities {capabilities} which involves communication between the following peers {peers}.</td>
</tr>
<tr>
<td>message</td>
<td>prov:Entity</td>
<td>The {subject} is a message that describes the a task that has the capabilities {capabilities} which involves communication between the following peers {peers}.</td>
</tr>
<tr>
<td>messaging action</td>
<td>prov:Activity</td>
<td>The {subject} is an activity that sends the message {message} between the following peers {peers}. The peer {peer} was responsible for sending the message to {peer}.</td>
</tr>
<tr>
<td>feedback report</td>
<td>prov:Entity</td>
<td>The {subject} is a feedback report, the feedback was left by the {peer/user/collective} about {agent/peer/user/collective/task/plan}. It was submitted using the {activity}.</td>
</tr>
<tr>
<td>reputation report</td>
<td>prov:Entity</td>
<td>The {subject} is a reputation report, it was derived from the feedback reports {feedback reports}. It was generated by the {activity} and was left by the {peer/user/collective}.</td>
</tr>
</tbody>
</table>

Template Sentences, where items in {} are variables which can be either a single item or a list.
Reputation Peer

<table>
<thead>
<tr>
<th>REST Method</th>
<th>URI</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET</td>
<td>/subject/byURI/:subject_uri</td>
<td>Retrieves feedback and reputation reports associated with a subject.</td>
</tr>
<tr>
<td>GET</td>
<td>/application/:app/subject/:subject/</td>
<td>Retrieves the set of subjects associated with an application.</td>
</tr>
<tr>
<td>GET</td>
<td>/application/:app/subject/:subject/feedback/</td>
<td>This call returns all feedback id’s about a subject.</td>
</tr>
<tr>
<td>POST</td>
<td>/application/:app/feedback/</td>
<td>Save a feedback report.</td>
</tr>
<tr>
<td>GET</td>
<td>/application/:app/feedback/:feedback/</td>
<td>Retrieves a feedback report with a specific id.</td>
</tr>
<tr>
<td>GET</td>
<td>/application/:app/subject/:subject/reputation/</td>
<td>Retrieves the set of reputation reports about a subject.</td>
</tr>
<tr>
<td>GET</td>
<td>/application/:app/reputation/:reputation/</td>
<td>Retrieves a reputation report with a specific id.</td>
</tr>
<tr>
<td>POST</td>
<td>/application/:app/opinions/</td>
<td>Retrieves the set of opinion reports about subjects.</td>
</tr>
</tbody>
</table>

Explantion Peer

| GET         | /application/:app/subject/:subject | Retrieve a narrative about a subject.                                     |

Provenance Store

| GET         | /store/api/v0/documents/ | Retrieve a list of documents visible to the authenticated user or public. |
| POST        | /store/api/v0/documents/ | Save a new document.                                                      |

Table 3
API calls for the reputation, explanation and provenance peers

<table>
<thead>
<tr>
<th>Category</th>
<th>Subcategory</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overall Star Rating</td>
<td>-</td>
<td>A five star rating for the ride</td>
</tr>
<tr>
<td>Ride</td>
<td>Price</td>
<td>A five star rating for the price of the ride</td>
</tr>
<tr>
<td></td>
<td>Route</td>
<td>A five star rating for the route of the ride</td>
</tr>
<tr>
<td></td>
<td>Timeliness</td>
<td>A five star rating indicating whether the commuter was ready on time</td>
</tr>
<tr>
<td>Individual</td>
<td>Reliability</td>
<td>A five star rating indicating the reliability of the commuter.</td>
</tr>
<tr>
<td></td>
<td>Communication</td>
<td>A five star rating indicating the communication during the ride.</td>
</tr>
<tr>
<td></td>
<td>Friendliness</td>
<td>A five star rating indicating the commuter’s friendliness.</td>
</tr>
</tbody>
</table>

Table 4
Categories and Descriptions for User Feedback.

work is to explain to users how their ratings are computed; over time, there will be an investigate into more complex functions to compute reputation category values.

5.2. Ride Share Vocabulary

In addition to the vocabulary defined for Smart City applications, the following subtypes for Ride Share are used and the hierarchy of the model is shown in Table 6. This vocabulary focuses on describing the activities in Ride Share. Providing precise typing about these different activities then enables the explanation peer to provide individualised descriptions of Ride Share activities. The namespace used for the vocabulary is http://smartsociety-project.github.io/cas/ and it defines the following terms:

1. A **driver** is a user that has a role as a driver.
2. A **rider** is a user that has a role as a rider.
3. A **ride_feedback_report** is a feedback report that pertains to a ride.
4. A **submitting_feedback** is an activity that submits a feedback report to the reputation peer.
5. A **storing_feedback** is an activity used by reputation peer to store a feedback report within it.
6. A **ride_request** is a task that describes a ride request and contains information about a proposed ride including, date and time, origin and destination, and the role of the user submitting the task.
7. A **reputation_item** is a JSON object that is returned from a GET request to the reputation peer.
8. A **authenticating_activity** is an activity which authenticates a user.
<table>
<thead>
<tr>
<th>Category</th>
<th>Subcategory</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average Overall Star Rating</td>
<td></td>
<td>average(overall_star_rating)</td>
</tr>
<tr>
<td>Ride</td>
<td>Average Price</td>
<td>average(price)</td>
</tr>
<tr>
<td></td>
<td>Average Route</td>
<td>average(route)</td>
</tr>
<tr>
<td></td>
<td>Average Timeliness</td>
<td>average(timeliness)</td>
</tr>
<tr>
<td>Individual</td>
<td>Average Reliability</td>
<td>average(reliability)</td>
</tr>
<tr>
<td></td>
<td>Average Communication</td>
<td>average(communication)</td>
</tr>
<tr>
<td></td>
<td>Average Friendliness</td>
<td>average(friendliness)</td>
</tr>
</tbody>
</table>

Table 5
Reputation categories and formula for summary driver and commuter reports

9. **storing_task** is an activity that stores a task locally.
10. **computing_composition** is an activity that computes a set of valid tasks given constraints or negotiation inputs.
11. **computing_task_complement** is an activity that identifies which set of tasks that are no longer valid.
12. **sending_request** is an activity that sends a request to peer.
13. **sending_negotiation_response** is an entity that contains the response to a negotiation activity.
14. **posting_task_request** is an activity that posts a task to orchestration peer.
15. **changing_view** is an activity that changes the view in a UI.
16. **composing_activity** is an activity that may be comprised of the following activities authenticate, compute_composition and compute_task_complement.
17. **negotiating_activity** is an activity which submits a task that may be used to modify another task, it may comprise of an authenticate and storing_task activities.
18. **submitting_activity** is an activity which submits a task, it may comprise of an authenticate and storing_task activities.
19. **computing_reputation** is an activity that is run by the reputation peer when a feedback report is submit, it computes a reputation report for all the subjects referred to by the submitted feedback report.

7. **Sentence Templates for Ride Share**

The Ride Share vocabulary enables the explanation peer to tailor the sentences it generates, with descriptions of the specific activities in Ride Share. Table 6 details the sentence templates exploiting the Ride Share vocabulary, similar to Table 2 it does not make the provenance paths used to bind the variables explicit because of conciseness, and instead uses the possible variable’s types for the bindings.

The explanation peer is used to explain the provenance data stored by Ride Share to its users. Specifically, it can describe:

1. The generation of reputation and opinions reports;
2. The generation of ride plans;
3. The negotiation process among users for selecting ride plans.

<table>
<thead>
<tr>
<th>Prov type</th>
<th>Smart City type</th>
<th>Ride Share type</th>
</tr>
</thead>
<tbody>
<tr>
<td>prov:Entity</td>
<td>entity</td>
<td>ride_feedback_report</td>
</tr>
<tr>
<td>prov:Activity</td>
<td>activity</td>
<td>submitting_feedback</td>
</tr>
<tr>
<td>prov:Activity</td>
<td>activity</td>
<td>storing_feedback</td>
</tr>
<tr>
<td>prov:Activity</td>
<td>activity</td>
<td>computing_reputation</td>
</tr>
<tr>
<td>prov:Activity</td>
<td>activity</td>
<td>composing_activity</td>
</tr>
<tr>
<td>prov:Activity</td>
<td>activity</td>
<td>authenticating_activity</td>
</tr>
<tr>
<td>prov:Activity</td>
<td>activity</td>
<td>storing_task</td>
</tr>
<tr>
<td>prov:Activity</td>
<td>activity</td>
<td>computing_task_complement</td>
</tr>
<tr>
<td>prov:Activity</td>
<td>activity</td>
<td>submitting_activity</td>
</tr>
<tr>
<td>prov:Activity</td>
<td>activity</td>
<td>sending_negotiation_response</td>
</tr>
<tr>
<td>prov:Activity</td>
<td>activity</td>
<td>negotiating_activity</td>
</tr>
<tr>
<td>prov:Activity</td>
<td>activity</td>
<td>posting_task_request</td>
</tr>
<tr>
<td>prov:Activity</td>
<td>activity</td>
<td>sending_request</td>
</tr>
<tr>
<td>prov:Activity</td>
<td>activity</td>
<td>changing_view</td>
</tr>
<tr>
<td>prov:Entity</td>
<td>plan</td>
<td>ride_plan</td>
</tr>
</tbody>
</table>

Table 6
The hierarchy of the Ride Share vocabulary.
Table 7

<table>
<thead>
<tr>
<th>Id</th>
<th>Ride Share type</th>
<th>Sentences</th>
</tr>
</thead>
<tbody>
<tr>
<td>RS1</td>
<td>reputation_report</td>
<td>The [subject] is a reputation report. It was generated by the [computing_reputation] activity and by the [peer].</td>
</tr>
<tr>
<td>RS2</td>
<td>submitting_feedback</td>
<td>The submit feedback activity [subject] was performed by the [peer]. This activity submitted the feedback [ride_feedback_report] to the [reputation_peer].</td>
</tr>
<tr>
<td>RS3</td>
<td>storing_feedback</td>
<td>The store feedback activity [subject] was performed by [reputation_peer]. This activity was used to store the feedback [ride_feedback_report] in the [reputation_peer]'s store.</td>
</tr>
<tr>
<td>RS4</td>
<td>computing_reputation</td>
<td>The compute reputation activity [subject] was performed by [reputation_peer]. This activity generated the reputation report [reputation_report] for the [user], and was derived from the following feedback reports [feedback_reports], it averaged the ratings for each of the categories in the feedback reports.</td>
</tr>
<tr>
<td>RS5</td>
<td>composing_activity</td>
<td>The composition activity [subject] was performed by [peer]. The composition activity is comprised of the following activities [computing_composition], [computing_task_complement] and [sending_response], it triggered the activity [sending_response].</td>
</tr>
<tr>
<td>RS6</td>
<td>authenticating_activity</td>
<td>The authenticate activity [subject] was performed by [peer], uses the [peer]'s identity to authenticate it.</td>
</tr>
<tr>
<td>RS7</td>
<td>storing_task</td>
<td>The store task activity [subject] was performed by [peer], and it stored the task [task] in its store.</td>
</tr>
<tr>
<td>RS8</td>
<td>computing_composition</td>
<td>The compute composition activity [subject] was performed by [peer], it generates potential ride matches between current ride requests and matches on a requests origin, destination and time. The following ride requests [tasks] were matched using this activity.</td>
</tr>
<tr>
<td>RS9</td>
<td>computing_task_complement</td>
<td>The compute task complement activity [subject] was performed by [peer], it generates the complement set to potential ride matches and contains ride plans that are no longer valid or that weren’t feasible based on ride request’s origin, destination and time. The following rides were classified as complementary [task].</td>
</tr>
<tr>
<td>RS10</td>
<td>submitting_activity</td>
<td>The submit activity [subject] was performed by [peer], it is a composition of activities that include [authenticating_activity], a [computing_composition] and [storing_task], it triggered the activity [sending_response].</td>
</tr>
<tr>
<td>RS11</td>
<td>sending_negotiation_response</td>
<td>The sending negotiation response [subject] activity was performed by [peer] and it sent the [task].</td>
</tr>
<tr>
<td>RS12</td>
<td>negotiating_activity</td>
<td>The negotiation activity [subject] was performed by [peer]. The negotiation activity is comprised of the following activities [authenticating_activity], [computing_composition], [storing_task], it triggered the activity [sending_response].</td>
</tr>
<tr>
<td>RS13</td>
<td>posting_task_request</td>
<td>The post task request activity [subject] was performed by [peer].</td>
</tr>
<tr>
<td>RS14</td>
<td>sending_request</td>
<td>The send request request activity [subject] was performed by [peer], by a user with the identity [identity] via the [peer].</td>
</tr>
<tr>
<td>RS15</td>
<td>changing_view</td>
<td>The change view activity [subject] was performed by [peer], by a user with the identity [identity] via the [peer].</td>
</tr>
<tr>
<td>RS16</td>
<td>ride_plan</td>
<td>The ride plan [subject] was generated by the [computing_composition] activity. It details the attributes of a ride including its date and time, the origin and destination, and its potential participants and their roles. It was derived from the ride requests [ride_requests]. It was generated by the [activity].</td>
</tr>
<tr>
<td>RS17</td>
<td>ride_request</td>
<td>The ride request [subject] was submitted by [user] and it was generated by [sending_request]. It includes information about a requested rides date and time, the origin and destination, and the role played by the [user].</td>
</tr>
<tr>
<td>RS18</td>
<td>sending_response</td>
<td>The sending response task [subject] sends an acknowledgement to the [peer] to say that the [activity] has completed.</td>
</tr>
<tr>
<td>RS19</td>
<td>feedback_report</td>
<td>The feedback report [subject] was generated by the [submitting_feedback] activity. It contains the reputation left by [user].</td>
</tr>
</tbody>
</table>

Ride Share Template Sentences, where items in {} are variables which can be either a single item or a list.
6. Ride Share Accountability Use Cases

This section expands on the three use cases identified in Section 5.3 by providing examples of relevant provenance collected from the different peers within Ride Share. For each use case, the explanation peer to generate two narratives: the first narrative exploits the semantics provided by the Smart City/Smart Share vocabularies, whereas the second narrative simply relies on the domain-agnostic types and relations provided by PROV. Presenting these two narratives demonstrates the benefits of decorating provenance with application-specific semantics.

6.1. Accountability for Ride Plans

Ride Share generates ride plans when the ride matching peer receives a ride request. The provenance recorded for such ride plans includes descriptions about: which views the user viewed; the submission of the ride request via the UI; the ride matching peer storing and generating ride plans; and the request that the ride matching peer sends the reputation peer storing and generating ride plans; and the request for opinion reports. An example of the provenance data collected can be seen in Figure 5.

This use case is required to provide systems accountability for the generation of ride plans, therefore the ride plan rideplan:1 is used as the subject. The following narrative was generating using the semantics defined by the Smart City and Ride Share vocabularies:

Ride Share Narrative: The ride plan rideplan:1 was generated by the rideshareapp:composition-1 activity. It details the attributes of a ride including its date and time, the origin and destination, and its potential participants and their roles. It was derived from the ride requests with the ids 1 and 2. The ride request riderequest:1 was submitted by usr:ido. It includes information about a requested rides date and time, the origin and destination, and the role played by the usr:ido. The ride request riderequest:2 was submitted by usr:avi and it was generated by view:submitRequestButton. It includes information about a requested rides date and time, the origin and destination, and the role played by the usr:avi and it was generated by view:submitRequestButton. The compute composition activity rideshareapp:composition-1 was performed by rideshareapp:rsa, it generates potential ride matches between current ride requests and matches on a requests origin, destination and time. The following ride requests with the ids 1 and 2 were matched using this activity.

In contrast, the following narrative was generated about the same subject (rideplan:1) using sentence templates relying on the PROV semantics but ignoring the Ride Share types.

PROV Narrative: The rideplans:1 is a prov:Entity. Entity rideplans:1 was derived from riderequest with the ids 1 and 2. It was associated with agent/s rideshareapp: rsa. It was generated by the activity rideshareapp:composition-1. The riderequest:1 is a prov:Entity. It was associated with agent/s rideshareapp:rsa. It was generated by the activity view:submitRequestButton. The riderequest:2 is a PROV:entity. It was associated with agent/s rideshareapp:rsa. It was generated by the activity view:submitRequestButton.

In order to compare the two narratives, the sentences about the same provenance elements and relationships from both narratives are presented side by side (see Table 8). The table shows that the Ride Share Narrative provides additional information about activities in the form of summary description (see RS8 in Table 8). In contrast, the PROV Narrative was unable to provide this information, because this information not available via the provenance: the provenance only exposes who performed an active and any inputs or outputs. The narratives also differ when describing who is associated with the generating ride requests. The Ride Share Narrative describes the user submitting a request via an agent (see RS17 in Table 8). In contrast, the PROV Narrative only details the agent (see P5 in Table 8), because it could exploit more data from the provenance because of the specialised sentence templates. In general, the sentences in PROV Narrative lack clarity about the relationships and roles of the provenance elements, while the Ride Share Narrative was able to provide it because of the additional semantics in the Ride Share vocabulary.

6.2. Accountability for Negotiation Processes

The negotiation process among users for selecting ride plans requires a potential user agreeing to a particular ride plan. The provenance recorded during a negotiation includes descriptions about: which views the user viewed; the submission of the negotiation offer via the UI; the ride matching peer using the offer to
Fig. 5. The provenance data recorded from the submission of a ride request (for more details https://provenance.ecs.soton.ac.uk/store/documents/33414/).
Fig. 6. The provenance data recorded from the submission of a negotiation offer (for more details https://provenance.ecs.soton.ac.uk/store/documents/33415/)
Table 7 Id | Sentence | Table 1 Id | Sentence
---|---|---|---
RS16 | The ride plan rideplan:1 was generated by the rideshareapp:composition-1 activity. | P1, P6 | The rideplans:1 is a prov:Entity. It was attributed to rideshareapp:composition-1.
RS16 | It details the attributes of a ride including its date and time, the origin and destination, and its potential participants and their roles. | - | -
RS16 | It was derived from the ride requests with the ids 1 and 2. | P10 | Entity rideplans:1 was derived from riderequest with the ids 1 and 2.
RS17 | The ride request riderequest:1 was submitted by usr:ido and it was generated by the view:submitRequestButton. | P1, P5, P12 | The riderequest:1 is a prov:Entity. It was associated with agent/s rideshareapp:rsa. It was generated by the activity view:submitRequestButton.
RS17 | It includes information about a requested rides date and time, the origin and destination, and the role played by the usr:ido. | - | -
RS17 | The ride request riderequest:2 was submitted by usr:avi and it was generated by view:submitRequestButton. | P1, P5, P16 | The riderequest:2 is a prov:Entity. It was associated with agent/s rideshareapp:rsa. It was attributed to view:submitRequestButton.
RS17 | It includes information about a requested rides date and time, the origin and destination, and the role played by the usr:avi. | - | -
RS8 | The compute composition activity rideshareapp:composition-1 was performed by rideshareapp:rsa, it generates potential ride matches between current ride requests and matches on a requests origin, destination and time. The following ride requests with the ids 1 and 2 were matched using this activity. | - | -

Table 8
Comparison Table for Ride Plans Narratives

| Table 7 Id | Sentence | Table 1 Id | Sentence |
---|---|---|---|
RS16 | The ride plan rideplan:1 was generated by the rideshareapp:composition-1 activity. | P1, P6 | The rideplans:1 is a prov:Entity. It was attributed to rideshareapp:composition-1.
RS16 | It details the attributes of a ride including its date and time, the origin and destination, and its potential participants and their roles. | - | -
RS16 | It was derived from the ride requests with the ids 1 and 2. | P10 | Entity rideplans:1 was derived from riderequest with the ids 1 and 2.
RS17 | The ride request riderequest:1 was submitted by usr:ido and it was generated by the view:submitRequestButton. | P1, P5, P12 | The riderequest:1 is a prov:Entity. It was associated with agent/s rideshareapp:rsa. It was generated by the activity view:submitRequestButton.
RS17 | It includes information about a requested rides date and time, the origin and destination, and the role played by the usr:ido. | - | -
RS17 | The ride request riderequest:2 was submitted by usr:avi and it was generated by view:submitRequestButton. | P1, P5, P16 | The riderequest:2 is a prov:Entity. It was associated with agent/s rideshareapp:rsa. It was attributed to view:submitRequestButton.
RS17 | It includes information about a requested rides date and time, the origin and destination, and the role played by the usr:avi. | - | -
RS8 | The compute composition activity rideshareapp:composition-1 was performed by rideshareapp:rsa, it generates potential ride matches between current ride requests and matches on a requests origin, destination and time. The following ride requests with the ids 1 and 2 were matched using this activity. | - | -

generate valid and invalid ride plans. An example of the provenance data can be seen in Figure 6.

This use case is required to provide systems accountability about how Ride Share handles negotiations and how it uses the negotiation to generate valid and invalid ride plans. Therefore, an explanation of the negotiation rideshareapp:negotiation-1 is used as the explanation subject. The generated sentences are present in Table 9, sentences about the same provenance elements and relationships from both narratives are presented side by side. As in the previous use case the table shows that the Ride Share Narrative provides more detailed information about activities than the PROV Narrative. With the addition of the smart share vocabulary the Ride Share Narrative was aware that the negotiation was a composed of a set of activities, whereas the PROV Narrative could not leverage the provenance for this information because it is not expressed using PROV. Therefore the Ride Share Narrative could describe the set of activities that the PROV Narrative could not.

6.3. Accountability for Reputation Reports

Ride Share generates reputation and opinion reports when the reputation peer receives a feedback report. The provenance recorded for the submission of feedback includes descriptions about: which views the user viewed; the submission of the feedback report via the UI; the reputation peer storing and generating reputation and opinion reports. An example of the provenance data can be seen in Figure 7.

The generated sentences are present in Table 10, sentences about the same provenance elements and relationships from both narratives are presented side by side. As in the previous two use cases the table shows that the Ride Share Narrative provides more detailed information about activities than the PROV Narrative.

6.4. Analysis

The aim of the analysis is to contrast Ride Share Narratives and PROV Narratives, by identifying dis-
The negotiation activity \textit{rideshareapp:negotiation-1} was performed by \textit{rideshareapp:rsa}.

The \textit{rideshareapp:negotiation-1} was informed by \textit{rideshareapp:sendResponse-2}.

The \textit{rideshareapp:rsa} is a prov:Agent.

The sending response task \textit{rideshareapp:sendResponse-2} sends an acknowledgement to the \textit{rideserver:OpenRideServer-RS} to say that the \textit{rideshareapp:negotiation-1} has completed.

The \textit{rideshareapp:sendResponse-2} was a prov:Activity. It was associated with agent/s \textit{rideshareapp:rsa}. \textit{rideshareapp:negotiation-1} was informed by \textit{rideshareapp:sendResponse-2}.

The \textit{rideshareapp:rsa} is a prov:Agent.

The negotiation activity is comprised of the following activities \textit{rideshareapp:authentication-2}, \textit{rideshareapp:generateComposition-2}, \textit{rideshareapp:generateTaskComplement-2}, and \textit{rideshareapp:storeTask-1}.

The authenticate activity \textit{rideshareapp:authentication-2} was performed by \textit{rideshareapp:rsa}, uses the \textit{usr:ido}'s identity to authenticate it.

The compute composition activity \textit{rideshareapp:generateComposition-2} was performed by \textit{rideshareapp:rsa}, it generates potential ride matches between current ride requests and matches on a requests origin, destination and time. The following ride requests \textit{riderequest:1} were matched using this activity.

The compute task complement activity \textit{rideshareapp:generateTaskComplement-2} was performed by \textit{rideshareapp:rsa}, it generates the complement set to potential ride matches and contains ride plans that are no longer valid or that weren’t feasible based on ride request’s origin, destination and time. The following rides were classified as complementary \textit{ride plan:1} and \textit{ride plan:2}.

The store task activity \textit{rideshareapp:storeTask-1} was performed by \textit{rideshareapp:rsa}, and it stored the task \textit{rideshareapp:offers/t1} in its store.

\begin{table}[h]
\begin{center}
\begin{tabular}{|c|l|c|}
\hline
Table 7 Id & Sentence & Table 1 Id & Sentence \\
\hline
RS12 & The negotiation activity \textit{rideshareapp:negotiation-1} was performed by \textit{rideshareapp:rsa}. & P3, P5, P8 & The \textit{rideshareapp:negotiation-1} is a prov:Activity. It was associated with agent/s \textit{rideshareapp:rsa}. \textit{rideshareapp:negotiation-1} was informed by \textit{rideshareapp:sendResponse-2}. \\
\hline
- & - & P2 & The \textit{rideshareapp:rsa} is a prov:Agent. \\
\hline
RS18 & The sending response task \textit{rideshareapp:sendResponse-2} sends an acknowledgement to the \textit{rideserver:OpenRideServer-RS} to say that the \textit{rideshareapp:negotiation-1} has completed. & P3, P5, P12, P8 & The \textit{rideshareapp:sendResponse-2} was a prov:Activity. It was associated with agent/s \textit{rideshareapp:rsa}. \textit{rideshareapp:negotiation-1} was informed by \textit{rideshareapp:sendResponse-2}. \\
\hline
RS12 & The negotiation activity is comprised of the following activities \textit{rideshareapp:authentication-2}, \textit{rideshareapp:generateComposition-2}, \textit{rideshareapp:generateTaskComplement-2}, and \textit{rideshareapp:storeTask-1}. & - & - \\
\hline
R6 & The authenticate activity \textit{rideshareapp:authentication-2} was performed by \textit{rideshareapp:rsa}, uses the \textit{usr:ido}'s identity to authenticate it. & - & - \\
\hline
RS8 & The compute composition activity \textit{rideshareapp:generateComposition-2} was performed by \textit{rideshareapp:rsa}, it generates potential ride matches between current ride requests and matches on a requests origin, destination and time. The following ride requests \textit{riderequest:1} were matched using this activity. & - & - \\
\hline
RS9 & The compute task complement activity \textit{rideshareapp:generateTaskComplement-2} was performed by \textit{rideshareapp:rsa}, it generates the complement set to potential ride matches and contains ride plans that are no longer valid or that weren’t feasible based on ride request’s origin, destination and time. The following rides were classified as complementary \textit{ride plan:1} and \textit{ride plan:2}. & - & - \\
\hline
RS7 & The store task activity \textit{rideshareapp:storeTask-1} was performed by \textit{rideshareapp:rsa}, and it stored the task \textit{rideshareapp:offers/t1} in its store. & - & - \\
\hline
\end{tabular}
\end{center}
\caption{Comparison Table for Negotiation Processes Narratives}
\end{table}
Fig. 7. The provenance data recorded from a submitting feedback (for more details https://provenance.ecs.soton.ac.uk/store/documents/33416)
The ride_manager:application/1/reputation/1/ is a reputation report, it was derived from the feedback reports with ids 0, it averaged the ratings for each of the categories in the feedback reports.

The ride_manager:application/1/reputation/1/ is a prov:Entity. It was derived from feedback with the ids 0.

It was generated by the ride_manager:generate_reputation-1 activity and was left by usr:ido.

It was associated with agent/s rideshareapp: rsa. It was generated by the activity ride_manager:generate_reputation-1.

The compute reputation activity was performed by ride_manager:reputation_peer. This activity generated the reputation report ride_manager:application/1/reputation/1/, and was derived from the feedback reports with ids 1 and 2.

The feedback report ride_manager:application/1/feedback/0/ was generated by the ride_manager:storing_feedback-1 activity. It contains the reputation left by usr:ido.

The feedback:0 is a prov:Entity. It was associated with agent/s rideshareapp:rsa. It was generated by the activity view:submitFeedbackButton.

Table 10
Comparison Table for Reputation Reports

<table>
<thead>
<tr>
<th>use case</th>
<th>number of generic sentences</th>
<th>number of specialised sentences</th>
<th>number of resources exploited</th>
</tr>
</thead>
<tbody>
<tr>
<td>1: Ride Share Narrative</td>
<td>0</td>
<td>9</td>
<td>9</td>
</tr>
<tr>
<td>1: PROV Narrative</td>
<td>8</td>
<td>0</td>
<td>6</td>
</tr>
<tr>
<td>2: Ride Share Narrative</td>
<td>0</td>
<td>10</td>
<td>13</td>
</tr>
<tr>
<td>2: PROV Narrative</td>
<td>7</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>3: ride share narrative</td>
<td>0</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>3: PROV narrative</td>
<td>7</td>
<td>0</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 11
Summary statistics for the use case narratives

criminating characteristics of these narratives. To this end, sentences are categorised as either generic or specialised. Generic sentences are generated by making use of PROV concepts only, as per Table 1, whereas specialised sentences are those constructed with knowledge of Ride Share types, as per Table 7. Furthermore, given that the sentence template variables are placeholders for resources to be extracted from the provenance, and quantify the number of resources that each narrative exposes.

Table 11 shows that the explanation successfully uses the Smart City vocabulary and that the vocabulary has good coverage of the terms used in the provenance for Ride Share. Specifically, the Ride Share narratives all use specialised sentences, whereas the PROV narratives use generic sentences. Also because of the specialised semantics for the Ride Share application, the explanation peer was able to expose more of the resources that were relevant to the provenance subject. In more detail, the explanation peer could not identify all the activities that were related to composite activities in the second use case. Therefore, the Ride Share narrative exposed nine more resources than the PROV narrative. In the third use case, the PROV narratives uses more sentences to describe fewer resources, than the Ride Share narrative. This is because the sentence templates leveraged by the semantics in the Smart City vocabulary explicitly define associations between the provenance elements and therefore can explain more resources in fewer sentences.
7. Conclusion

The paper presents a vocabulary for Smart City applications that supports an accountable framework. The framework uses provenance to provide an account about how documents, data, and information in a Smart City application are used, modified, and created by both users and peers. The role of provenance and its semantic markup is critical to build accountable Smart City applications. The framework also provides users with narratives generated to support the transparency of automated processes. Specifically, this paper contributes to the state-of-the-art: A Smart City vocabulary specifically designed to support the markup of provenance to support accountability. The vocabulary is designed to support a board class of applications, however, it is beneficial to expand these terms to include information about how specific types of activities function; A framework that supports transparency and accountability using PROV, including the:

1. The reputation peer, a provenance enable reputation system that uses the Smart City vocabulary;
2. The explanation peer, which provides a narrative about a specified provenance element using the terms defined in the Smart City application vocabulary.

and, provenance explanation examples that demonstrate the benefits of using semantics to provide an account of provenance. The uses case discussed in Section 6 show that the explanation generated based on the Smart City vocabulary were able to exploit more data from the provenance, and could provide more specialised sentences describing a provenance subject.

This work will be extended in four ways, it will:
Firth, investigate how provenance data and the Smart City vocabulary can be extended or used to generate narratives for collectives, and how a narrative can be condensed while identifying commonalities and outliers in collectives. Second, support privacy by using provenance data that has been transformed to obfuscate and remove concepts that refer to sensitive information or are private because of privacy policies. Third, evaluate the explanation service’s narrative via a user study to validate how effect narrative to support decision making processes. Fourth, extend the json used in the serialisations so that they support json-DL.

8. Acknowledgments

The research leading to these results has received partially funding from the European Community’s Seventh Framework Program (FP7/2007-2013) under grant agreement n. 600854 Smart Society: hybrid and diversity-aware collective adaptive systems: where people meet machines to build smarter societies http://www.smart-society-project.eu/.

References


