Final Report

for

Project Title:

Study of India Appropriate Technology (IoT) Solutions for Smart Cities

April 2018
**Investigator Team**

**PI:** Dr. Bharadwaj Amrutur\(^1\), Professor, IISc Bangalore.

**Co-PI:** Dr. Hemant Darbari, ED, CDAC Pune.

**Contributors**

**From IISc:** Ashish Joglekar, Abhay Sharma, Vasanth Rajaraman, Rakshit Ramesh, Srikrishna Acharya, Yogesh Simmhan

**From CDAC:** Mr. Suresh V. (CDAC Pune) and Team

**Others:**

Mr. Arvind Tiwary (TIE), Mr. Alok Sethi (DMTS, Delhi) and team ([RFP Guidelines](#))

Ms. Pamela Kumar, (TSDSI), Mr. Bipin Kumar (Gaia)

Dr. Abhijit Lele (Robert Bosch Engineering, India), Mr. Sathya Sankaran (Mapunity), Mr. Krishanu Seal, Mr. Kishore Narang (Narnix), Dr. Prasant Misra (TCS), Dr. Geeta Manjunath (Xerox), Mr. Dileep Paruchuri (Intel), Dr. Narendra Nanjungud (Ericsson), Mr. Madhav Chablani (CCICI) and all the IOT4SCTF\(^2\) members.

**Acknowledgements**

We thank the **Ministry of Electronics and Information Technology**, Government of India, for supporting this study project. We also thank the Robert Bosch Centre for Cyber-Physical Systems (RBCCPS) at the Indian Institute of Science, Bangalore and Centre for Development of Advanced Computing, Pune for providing additional resources to undertake this study. We thank Subramani M. K. of RBCCPS for help with all the logistics.

---

1. amrutur@iisc.ac.in, Ph: +919886054723
2. https://sites.google.com/site/iot4sctf/
# Executive Summary

# IoT for Smart Cities Reference Architecture

## 1. Motivation and Context

## 2. Reference Architecture Perspectives

2.1. Communication Centric Perspective

2.2. Connectivity and IoT Device Management Perspective

2.3. Data Perspective

## 3. Reference Architecture for the Smart City Data Mart

3.1. Architecture Requirements

3.2. Components of the proposed middleware

3.2.1. Information Resource Catalogue

3.2.2. Data Schemas

3.2.3. Data Brokers

3.2.4. Data Stores

3.2.5. Video Broker and Store

3.2.6. Authentication, Authorization, Accounting, Auditing, Logging and Middleware Management

3.2.7. Policy Enforcement and data integrity checks

3.3. Data Middleware APIs

3.3.1. Registration APIs

3.3.2. Publish API

3.3.3. Subscribe API

3.3.4. Catalog API

3.3.5. Database API

## 4. Security aspects for IoT implementation for smart-cities

4.1. IoT Device Security

4.1.1. Default Username and Passwords

4.1.2. Principle of Least Privilege (Privilege Separation and Dropping)

4.1.3. Capabilities

4.1.4. Firewall and Intrusion Detection and Prevention Systems

4.1.5. Full Disk Encryption

4.1.6. Certificates, PUFs, TPM and Physical Security
9.2.3.2. Phased additions

9.3. Shared Infrastructure

9.4. Security

9.5. Open Data and Citizen services (Reimagined)
  9.5.1. Benchmarking
  9.5.2. Analytics + Big Data=Smart

9.6. Start-ups and MSME Inclusion

9.7. Citizen participation
  9.7.1. Crowdsourcing
  9.7.2. Feedback Loop

9.8. Interoperability, Standards and Reference Architecture
  9.8.1. Interoperability
  9.8.2. Device Management and Provisioning Standards
  9.8.3. Operations (OCC) or Network Control Centre (NOC)
  9.8.4. Common Components

10. RFP methods
  10.1. RFP toolkit
    10.1.1. Cheapest is not Best
    10.1.2. Quality cum Cost based selection (QCBS)
  10.2. Risk Reward

11. Annexure Open Data Portal
  11.1. Metrics

12. Annexure Security for IoT or Devices layer in Smart City
  12.1. Unique challenges for IoT Devices
  12.2. Minimal Recommendations
**Executive Summary**

The main objectives of our project are:

- Recommendation and Guidelines for Smart City RFPs related to IoT Specific technologies, driven by India specific use case analysis and emerging technology adoption methodology.
- India appropriate reference architecture for IoT enabled Smart Cities through a collaborative platform of domain experts from industry, academia, government, start-ups, professional bodies and user agencies.

We conducted two workshops with City administrators: one in Chandigarh and another in Electronics City, to understand the City’s requirements. A series of telecons/webcons were held with technical presentations from a diverse set of domain experts from industry, academia, govt. bodies, startups etc. Further deliberations and discussions were done to synthesize the view points in this report which are summarised below.

**Inputs from the City Administrators:**

1. Citizens must be the key beneficiaries of any smart city solution. It should benefit all strata of society.
2. Unified Command and Control centre is needed to be able to oversee all the city operations by the administrators.
3. Need low cost, sustainable, interoperable solutions with smartness built in to help with maintenance and operations.
4. Solutions should be vendor technology (software or hardware) agnostic – every component should be replaceable.
5. Solutions should enable development of future smarter/intelligent solutions.

**RFP Guidelines**

1. Should explore use of non-licensed sub-gigahertz spectrum for LPWAN, including non-IP based last mile solutions (BLE, Zigbee, LoRA, VLC etc).
2. Since Smart City solutions will take years to deploy and operationalize, and with rapid technology advancements, costs come down, RFP should incentivize providers to reduce their price over time via rebasing their technology and enable phased additions of technology over time.
3. Since light poles are ubiquitous in cities, they should be used as key anchor points for smart city solutions
4. MoUD’s guidelines on Cyber Security Model Framework for Smart Cities provides best practices framework. In addition, a central cell under NCIIPC (National Critical Information Infrastructure Protection Centre) to develop methods to test
Smart infrastructure (whitelist hackers) and arrange emergency response in situations of attack can be set up.

5. Open data will be critical for further empowerment of citizens as well as emergence of new services. The RFP should clearly ask for data in open formats to be made available to the city by the vendors of various solutions.

6. RFPs can give an opportunity to MSMEs and startups to spur the local ecosystem. These can play a major role in training/maintenance as well as offer novel low cost solutions in the case of start-ups.

7. Incorporate crowd sourced data to enhance citizen participation via enabling them to report issues etc.

8. RFPs should promote interoperability of solutions from different vendors.

9. Device management will be critical and a standards-based approach will be essential for this.

10. Quality cum cost based system can be adopted to ensure high quality smart city solutions.

**India Appropriate Reference Architecture for IoT Enabled Smart Cities**

1. **Enabling easy collection and exchange of data is a key enabler for future smart city applications.** We advocate an hourglass architecture like in AADHAR, with the neck of the hourglass being the middleware that mediates between data producers (IoT devices, Sensors, existing IT systems etc.) and consumers (applications like the unified command and control centre, city business decision support etc.).
2. Providing good, well documented, open APIs for the Smart City middleware, along with standardising the data schemas will be key to enable a rich application and device ecosystem to evolve.

3. A data middleware should be implemented, that will decouple sensors/IoT devices from the applications. This will allow 3rd party applications to be developed for existing IoT devices. It will also allow IoT devices from multiple vendors to be plugged in an interoperable manner.

4. Open data models for all IoT data sources (and actuators) needs to be developed to enable interoperability and vendor neutrality of devices in the IoT system. In addition, the models should be extensible, allowing for easy enhancement; and operations need to be developed which will seamlessly integrate the enhanced information model with existing applications.

5. India appropriate city information model (ontology) for Indian cities, their structures and their operations need to be developed which will enable future smart/intelligent solutions.

6. A data/sensor catalogue should be provided which is machine and human searchable via standard REST API. The catalogue will contain information of all available data sources (IoT sensors, their capabilities, owners, access methods etc.) in a standard form using open templates like JSON.

7. The data middleware should provide the capability to subscribe to live data as well as access archival data from the IoT devices via standard REST and Streaming APIs.

8. The data middleware should support AAAA (Authentication, Authorization, Accounting and Auditing) capabilities for devices, applications and human users. It will support both real and virtual devices (sensors). It should support complete lifecycle management of devices.

9. The AAAA should implement state of the art security features in compliance with MoUD recommendations and extend it further as required. Security certificates, API keys, authentication tokens, etc. should be managed by the AAA. Support for SLAs and federated authentication should be provided.

10. Authorization capability should enable fine grained access control to resources. Coupled with accounting and interface to digital payments infrastructure, this will enable developing a data marketplace.

11. The data middleware’s interfaces and data formats will be “open” allowing for easy replacement of any component, migration to new system integrators, migration to new platforms etc. without losing valuable data or incurring too much downtime.

12. A standards-based management for devices, equipments and applications,

---

should be adopted to manage the ICT components of the smart city implementations. This will enable multiple vendors to offer device, ICT and solution management services. Device management should be decoupled from ICT and solution management as well as the data services and applications.

**Suggestions for the future**

1. Since there is not much experience with Smart City solutions in India, we recommend the setting up of sand-boxes/living labs/test-beds in every city to enable experimentation, learning and teaching of smart city concepts in these spaces. These can be co-located with a local university and involve local MSMEs and other companies, in collaboration with city administration.

2. The Smart City test-bed will enable exploration of new concepts by researchers, demonstration of new technologies by entrepreneurs, teaching of Smart City concepts to city employees and students and spur the development of local solutions to local problems.

3. Every Smart City can be encouraged to set aside a small budget (<1%) for the Smart City test-bed, with additional support from state and central R&D funds.

4. Interoperability and compliance test centres and frameworks for Smart City solutions should be set up.

5. Extensible, open source technologies and standards should be adopted/developed wherever possible to enable low cost/sustainable/vendor neutral solutions.

6. Common, well documented APIs, and data schemas need to be developed and adopted across all smart city implementations in India. This activity can emerge from test beds and needs to be embarked upon immediately. A consortium of implementers should be formed to drive this activity.

7. City information model needs to be developed. Such information models need to be distributed and extensible. This will be essential for emergence of smart/intelligent applications of the future. This has to be driven by one of the Indian standards bodies.

8. City, region & domain specific guidelines/standards in practically every sphere are needed. For example, smart street lighting standards in terms of minimum lux for safe city and hence the pole heights, LED lumens, installation guidelines, backup procedures etc. are needed for smart street lighting. Similar standards are required for surveillance (camera resolutions, deployment guidelines etc.) and practically every other application area. Indian standards bodies should take these up one at a time to develop a thorough standard and reference implementation for each domain.

**Gaps in IoT technologies for Smart Cities for future R&D**

1. Low cost smart IOT devices for many city applications, customised to each city's requirements, along with open data models are needed. For example, multi-color
LEDs for North India, to effectively deal with fog in winter months.
2. Distributed trust technologies for audits (for e.g. Blockchain) should be explored for use in Smart City services.
3. New analytics for video will be needed to enable more effective use of cameras.
4. Distributed/edge analytics need to be explored, especially for video.
5. High priority data channels for emergency services for handling critical data and its model, like gas leakage, fire, etc.
6. Indian City specific information models/ontologies are missing and need to be developed. These will be essential for future “Intelligent” applications.
IoT for Smart Cities Reference Architecture

1. Motivation and Context

It is now well recognized that “data is the new oil”\(^4\). Organizations that can use data effectively, are more efficient, nimble and productive. Cities contain/generate vast troves of data that can provide immense benefit and value to all the stakeholders. Hence the city’s IT systems should enable easy production and dissemination of data to stakeholders and value creators.

IoT (Internet of Things) is a new technology that has emerged over the last few years and it promises a further explosion of data sources. This technology will essentially allow any “thing” to start producing and communicating data. For instance, every streetlight, lamp post, garbage bin, transformer, etc. – almost anything one chooses, can be made to continuously report their status, their observations, their usage patterns, etc. This data, in turn, can be used by downstream business/administration processes to provide timely, efficient, value-added services. In this report, we restrict ourselves to architectural constructs to incorporate these new IoT based systems in Smart City infrastructure.

One of the key lacuna in the current IT systems in Indian administrative organizations is the lack of easy access to data by entities outside the organization, especially for machines/applications. Though data is plentiful, it is either locked up in proprietary formats or unavailable due to lack of APIs. In this report, we look at architectural constructs which will facilitate easy exchange of data (regulatory/policy issues are also important to be addressed, but beyond the scope of this study). Hence we take a data/information centric view of the reference architecture as opposed to an application/communication centric view.

It has been well recognized that data/information connectivity will be key to enable the emergence of new smart applications and its main ingredient is the concept of linked data\(^5\). What this entails is that there should be some additional data about the data – called meta-data, that can place the data in a broader context. For example, an air quality monitor might give a series of numbers of the CO2 level. However, if that number stream is annotated with additional information like the location of the device, the make of the device, the units of the reading, the time samples, etc., it allows for a richer and more meaningful interpretation of the data stream especially by new users. While it is not challenging technologically to provide this meta-data, in practice there are no agreed upon standards nor requirements to provide this. Hence a key aspect of our study is to look at possible data formats (or schemas) which could be used for IoT

\(^5\) Tim Berners-Lee, Linked Data https://www.w3.org/DesignIssues/LinkedData.html
devices and which incorporate additional meta-data that will enable more effective use of the data. If device vendors agree to use the same data schema, then this will go a long way in enabling interoperability of vendor devices in the Smart City infrastructure.

The ongoing implementations of Smart City solutions, have tended to be “silod”, i.e. each application has an end-to-end implementation that stands alone and doesn’t interact with other applications (Figure 2).

![Figure 2: Vertical approach for supporting smart city use cases.](image)

In fact, within the same application domain, like smart lighting for instance, multiple vendors are contracted to provide solutions to different city areas and these too don’t interoperate. While this approach leads to quick rollout of solutions for specific use cases, in the long run it will prove to be inefficient and suboptimal. More specifically, we see the following key problems with this approach:

1. The end IoT devices can only interact with specific vendor software stack. This kills the interoperability of the IoT devices. While a “dumb” bulb from any number of different vendors could be used before, now one could get locked into a “smart” bulb from only a single vendor. This is mainly due to the lack of an agreed upon data schema/data model for the IoT devices.
2. The data from the devices is locked up in the vendor system in proprietary formats. Hence it is not easy for the owner (city) to share or reuse the data for other purposes – or even monetize it later.
3. The application for visualization/dashboarding and control of the devices are tightly coupled to the devices. For instance, it will not be easy for a 3rd party to develop a control application for the same IoT devices.
4. An analysis of the use cases of the Indian smart cities (Appendix A) will indicate that some of the sensors – especially the cameras, can be used across several different applications, provided the right analytics are developed. With the current implementation approach, this is not easily doable – unless the same vendor takes up this task.
5. One of the key resources for the cities will be data. Easy access to data will of course be of great benefit to the city administration itself in terms of adopting “data driven policy and decision making”. In addition, easy access to data could spur new innovations and novel applications that will benefit the citizens and the
economy. Finally, the cities could look to monetize this data – leading to a way to offset the costs of deploying smart infrastructure. With the existing approaches, the ability to easily exchange data in a safe secure manner is missing. This report will articulate an architecture/approach which will help to address all the problems mentioned above.

2. Reference Architecture Perspectives

There are multiple perspectives to the Smart City Architecture, each of which highlight a different aspect of the system.

2.1. Communication Centric Perspective

In the communications centric perspective shown in Figure 2, different communication technologies are needed to connect the various sensors/devices to the ICT framework.

![Communications Centric Perspective](image)

**Figure 3: Communications Centric Perspective**

We have wireless and wired technologies, as well as IP (internet protocol) and Non-IP technologies. The cellular based wireless technologies (GSM/3G/4G) are the most widespread and rely on licensed spectrum and provide good service, large area coverage (~few kms), high data rates, yet suffer from large power requirements on the device side. However newer standards like LTE-M1 and NB-IoT promise to improve upon the power shortcomings, preserve the large area coverage, but offer low data rates suitable for sensors (~few kbit/s). WiFi based technologies allow high data-rates over short distances and are IP compatible. Passive Optical Networks (PON), specifically GPON (Gigabit PON) use optical fibres to provide high bandwidths (~few Mbits/s). These are IP compatible and are excellent for city wide high bandwidth backbone
infrastructure - especially to support IP cameras. Wireless networks like BLE (Bluetooth Low Energy) and Zigbee use unlicensed spectrum, cover short distances (<5-30m) and hence are low cost, but also low bandwidth and low power. These are non-IP based and need interfaces to convert to the IP network. Long range, low power wide area networks (LPWAN), on the other hand, provide coverage similar to cellular networks, yet use unlicensed spectrum (e.g. LoRaWAN) are also non-IP based. Finally, many industrial type installations (like in power plants, water pump houses, etc.) might use wired network infrastructure, which are non-IP based. In general, one can expect a heterogeneous mix of communication technologies and the city’s smart city infrastructure should be prepared to accept new and emerging technologies, as long as they deliver value in terms of cost and performance.

2.2. Connectivity and IoT Device Management Perspective

![Figure 4: Connectivity Centric Perspective](image)

In this perspective, the low level communications technologies are abstracted out and the devices can be reachable either via IP or non-IP technologies. Usually, the non-IP connectivity technologies are restricted to the leaf of the network – to connect to the final sensors/actuator devices using communication technologies like BLE, LoRa, SCADA networks, in-computer wired networks, etc. An important characteristic of being IP compatible is that the powerful security/management features can be easily incorporated into these devices – as there is a world-wide community, both commercial as well as open source, that is working on these. However, for various reasons (like limited resources or proprietary/legacy networks) it becomes necessary to use non-IP devices, in which case a gateway node (usually linux based) is necessary to connect these to the IP network and hence the ICT middleware. Many existing and/or legacy IT systems should also be brought online and this can be achieved via adaptors (which are also a form of gateway), which are pieces of software running on servers. Smartphones being one of the most ubiquitous sensor/human interface devices are special as they
integrate the gateway/sensor in the same hardware and are usually android or iOS based. From a IoT device management perspective, device and gateway lifecycle management (from first time connection to maintenance to removal), security credentials provisioning and maintenance (first time security keys, certificates and periodic updates), firmware and software updates (security patches, end-application updates, etc.), device and gateway health monitoring become important tasks for the management infrastructure. With ever evolving cyber-threats, it becomes important to be able to patch and upgrade the softwares in the gateways on a regular basis. There are many existing and emerging standards and solutions to cater to this perspective, with many industry alliances (like Open Mobile Alliance, Broadband Forum, etc.) developing standards and frameworks. Smart City solutions should adopt best-in-class, industry standard solutions to solve these problems.

2.3. Data Perspective

At an even higher layer of abstraction sits the data perspective, which captures the ultimate interest in data for the various stakeholders in the cities, as the data can be used to derive benefits for their day to day operations and lives. In this perspective (shown in figure 5), we can classify two kinds of entities - data producers and data consumers.

Data consumers could be applications like integrated command and control centers, various management applications like street lighting management, traffic light management, etc., systems, and citizens (via their smartphones/desktops). Data producers will be IoT devices (like air quality sensors, cameras, lights, etc.), legacy ICT systems (like police IT, various government department IT systems) and people themselves. The city’s data layer sits in between the two – as depicted in Figure 5. Note that the same entity can act both as a data consumer and a data producer. For the rest of this document, we will focus on the architecture of this perspective as this is the least developed from a standards point of view in the current scenario, and yet is the most
critical for fully realizing the potential of “Smart Cities”.

3. Reference Architecture for the Smart City Data Mart

3.1. Architecture Requirements

We first outline the desirable properties/requirements of such a middleware from a functional/operational perspective.

We use the standard term - resource to describe any piece of data/information/service hosted in the middleware.

*(Req1)* The middleware should support any data producing entity (IoT device, citizen as a sensors, virtual sensors etc) and support any data consuming entity, agnostic to specific verticals/domains.

This is essential to allow the emergence of new cross-silo applications.

*(Req2)* All accesses to the middleware should be enabled via well-designed and documented APIs (application programming interfaces), over well-known application protocols, following standard best-practices for security. These APIs should be city-agnostic and hence the same apps should work in any Indian city (with appropriate security credentials).

Most Indian cities share similar processes and hence applications for one city should be easily portable to others. This will drive down development cost as well as improve market reach. This can be only achieved if the APIs to the city middleware are standardized (which might be a tall order). From a best-practises perspective, the APIs should be REST-based and also include streaming APIs for access to continuously changing data. The APIs should be supported over well-known application protocols like HTTPS, AMQP, MQTT, long-lived HTTP, websockets, etc. and incorporate standard security schemes like (SSL, TLS, Certificates, etc).

*(Req3)* The owner of the middleware should be able to create accounts, provide identity credentials and define access rights to the users (humans, organizations and applications) of the middleware.

The users could be individual citizens or organizations – essentially legally defined entities. The users will employ software entities (one or more), which interact with the middleware on their behalf. While an individual citizen will likely interact using a mobile app or a browser, an organization could have many devices or apps that could publish data to or pull data from the middleware. The middleware owner should also be able to define the access rights of the users (or their software agents).

These will include the following:

- The ability to define the set of resources the user can access.
- The set of resources they can create/update.
- Limits on the amount of data/data-rate they are permitted.
- The time validity of their access rights.
- And support new access policies that might emerge in the future.

(Req4) Authenticated and authorized Users/Applications should be able to access live and archival data related to a sensor/device. Such Users/applications should also be able to easily upload/publish data to the middleware, provided they are authenticated and authorized.

Besides accessing live information, the users should be able to access historical information. Authenticated users should be able to easily share/publish data to the middleware. This will enable quick on-boarding of new data streams from devices installed by city administration, and enable crowd sourced data and data from other organizations to be on-boarded quickly and easily. Commercial value-added services based around data (like data curation, annotation, analytics etc.), by 3rd parties should be easily supported by the middleware.

(Req5) There should be a catalogue of available resources and services. The catalogue should be both human and machine readable. The catalogue should indicate all the necessary information about the resources, including their meta-data (i.e. data schema or data model), as well as a how to access them.

The catalogue of resources is analogous to the online shopping catalogues from online retail sites. The consumer can browse through available products, their features, prices, reviews etc. and then decides to purchase a subscription or a copy of them. The city’s middleware should provide this facility as this will nucleate the development of a new data/digital economy.

(Req6) It should be possible to on-board information onto the middleware from third party and legacy systems.

This will be doable via writing of suitable “adaptors” which will marshal information between the legacy systems and the middleware. For example, the city administration might want to get a high-level picture of number of failed smart water meters, say monthly, as opposed to detailed information about these failures as that will be the responsibility of the water supply boards. This information can be pushed from the water-board’s systems into the middleware for consumption by the city officials.

(Req7) The architecture/middleware should be agnostic to specific implementation choices for the components, but be adaptable to incorporate emerging technological advances.

For example, it should be possible to use a variety of database technologies to store the data. The format and schema of the data will be published and hence it should be possible to migrate the data across new database technologies. The same applies to all the other components. The data and API schemas, the authentication technologies etc. should all be modifiable/extensible in the future. This is very important as the
technologies themselves are undergoing rapid evolution and it is important for the systems to be able to keep up and adapt.

3.2. Components of the proposed middleware

In contrast to the ongoing deployments, which are based on the vertical architecture (figure 2), we advocate a horizontal architecture for the IoT system to allow for “non-siloed” solutions (figure 6). This is agnostic to specific applications – and in fact should support any device and any applications. One of the key questions is to identify the main components and their interactions/interconnections (i.e. the architecture) of such a system. Here one needs to perform a delicate balancing act – the more things that are specified, the less scope there is for innovation. At the same time, specifying too little will lead to chaos and that too deter innovation. To help us move forward, we take inspiration from the Aadhar architecture, which advocates an hourglass architecture. The neck of the hourglass is the key middleware which if chosen carefully and minimally, will enable a whole host of new applications and use cases.

The middleware essentially mediates between IoT data producers (real and virtual sensors, devices, legacy systems etc.) and IoT data consumers (e.g. applications like the Unified Command and Control Centre, city business decision support etc.). APIs for interactions with the middleware should be well-documented and based on standard application protocols and best practices. API endpoints for accessing the data and/or services should be searchable via an open catalogue, and should contain linked metadata about the resources.

The middleware will enable decoupling of sensors/IoT devices from the applications. This will allow 3rd party applications to be developed on existing IoT devices. It will also allow IoT devices from multiple vendors to be plugged in an interoperable manner.

Figure 6 outlines the key components of the middleware. Here we don’t imply any implementation or hosting model (example monolithic vs. micro-services, local versus cloud versus SaaS, etc.), but just point out the key components and the need for APIs to access the various services.

---

6 https://www.slideshare.net/AmitRanjan/aadhaar-technology-architecture
3.2.1. Information Resource Catalogue

The information resource catalog is one of the key components of the Smart City data middleware. It contains a list of data resources and their descriptions, including API endpoints, and other meta-data like access hints, ownership, providers, list of parameters and their descriptions etc. A useful analogy is the online shopping catalog, where a consumer can browse through available products, their features, reviews etc. and then decide to purchase a subscription or a copy of them. The resource catalog plays a similar role for the clients wishing to use the smart city resources and, as additional benefit, it can help nucleate the development of a new data/digital economy.

As an example, Hypercat is a recent alliance which has developed a standard for a catalogue under the British Standards Institution (PAS212)\(^7\). It is a lightweight, JSON based hypermedia catalogue format for exposing a collection of Uniform Resource Identifiers (URI) for IoT assets. The interactions with the catalogue use REST APIs over HTTP and the catalogue items are stored as RDF-like triples. RDF triples are essentially of the form “Subject, Predicate, Object” and can be used to encode relationships and hence provide a way to link the data/device parameters to external concepts. This will enable creating appropriate metadata for the devices that can then provide the right context and semantics for analysing the data from the IoT device. The framework allows a lot of flexibility in defining new annotations for new devices or information sources. In addition, Hypercat also advocates a few different search APIs, with a key one based on

\(^7\)http://shop.bsigroup.com/forms/PASs/PAS-212-2016-download/
geographic indices. Another important feature in Hypercat is the possibility of having links to other catalogues. This is to enable federated/distributed growth of IoT resources which will be a very desirable property.

### 3.2.2. Data Schemas

*Open data schemas from all IoT data sources (and actuators) needs to be developed to enable interoperability and vendor neutrality of devices in the IoT system. In addition, the schema needs to be extensible, allowing for easy enhancement.*

There are many online repositories for schemas like json-schema.org (for schema templates in JSON), or www.oneiota.org for some data schemas for some IoT devices. In addition, projects like FiWare have also proposed schemas⁸. We need to adapt and extend existing schemas (and develop new ones if they do not exist), to meet the needs for the IoT devices (and other data resources) for the Indian Smart Cities. As an example, we have discussed (and made publicly available⁹) examples of schemas for street light and power meter.

Recently, the Open Connectivity Foundation¹⁰, which is an international consortium of companies and some academic institutions, has developed data schemas for many IoT devices¹¹. The schemas are based on “JSON schemas”¹². We find these to be a better way to describe the metadata than that proposed in Hypercat as there are open-source tools readily available that allow schema creation, integrity checking of the schemas, etc. (Essentially, JSON-schema allows a simple grammar to be defined and hence it is easy to check schemas against that). This will be particularly important when one wants “user-contributed” schema development and deployment.

We recommend combining the best features of Hypercat (API-based access to catalogue, with powerful search capabilities), and OCF (JSON schema-based information description to enable integrity checking). We believe that the structures proposed in OCF can be further extended as described next.

As in OCF, we recommend to use JSON schemas to define the structure of each entry in the resource catalog. The schemas define specific fields to be included in the metadata for a given class of devices. Apart from mandating the inclusion of certain fields, there exist provisions to include (optional) device/vendor specific fields, e.g., links to reference ontologies, additional device information, usage hints, etc. which enhances the usability of data by third party applications. The schemas are further used to validate the catalog entries at the time of on-boarding the device using the easily available json-schema validation tools¹³.

---

⁸ https://www.fiware.org/data-models/
⁹ https://github.com/rbccps-iisc/smart cities schemas
¹⁰ https://openconnectivity.org/
¹¹ https://oneiota.org/
¹² http://json-schema.org/
¹³ http://json-schema.org/implementations.html
Towards defining the structure of catalog items, we propose to organize the metadata into following categories:

A. Static information: General information about the device, e.g., geolocation, type of device, ID, tags, etc.

B. Observation sub-schema: Meta-information for the observations made by the device, e.g., sensed parameters, etc.

C. Control sub-schema: Meta-information for the actuation parameters accepted by the device, e.g., control inputs, etc.

D. Configuration sub-schema: Meta-information for the configuration parameters of the device, e.g., sampling rates, sleep times, etc.

E. Management sub-schema: Meta-information related to device management.

Of course the above could be further extended in the future.

A key idea is to store observation/control/configuration information within a catalog entry as a JSON schema itself. Thus, the metadata becomes an actionable specification that can be used to dynamically validate the data being sent from/to the device (see Figure 7).

**Figure 7:** A device schema in the catalog is validated against a global-meta-schema. Each data item coming from/to the device is further validated by the embedded schemas in the device schema. /cat is the API endpoint for the Information Resource Catalog in the middleware.
We give an example of a schema entry in the catalog for a smart streetlight.

```json
{
    "refCatalogueSchema": "generic_iotdevice_schema.json",
    "id": "70b3d58ff0031de5",
    "resourceType": "streetlight",
    "tags": [
        "onstreet",
        "Energy",
        "still under development!"
    ],
    "refCatalogueSchemaRelease": "0.1.0",
    "latitude": {
        "value": 13.0143335,
        "ontologyRef": "http://www.w3.org/2003/01/geo/wgs84_pos#"
    },
    "longitude": {
        "value": 77.5678424,
        "ontologyRef": "http://www.w3.org/2003/01/geo/wgs84_pos#"
    },
    "owner": {
        "name": "IISC",
        "website": "http://www.iisc.ac.in"
    },
    "provider": {
        "name": "Robert Bosch Centre for Cyber Physical Systems, IISc",
        "website": "http://rbccps.org"
    },
    "geoLocation": {
        "address": "80 ft Road, Bangalore, 560012"
    },
    "accessMechanism": {
        "requestAccessSite": {
            "describes": "URI for getting permissions to access the device",
            "value": "http://rbccps.org/middleware/requestAccess"
        },
        "accessEndPoint": {
            "value": "https://rbccps.org/middleware/api/[api_ver]/db",
            "describes": "End point to access the archived values (database access endpoint)"
        },
        "subscriptionEndPoint": {
            "value": "mqtt://rbccps.org/subscription/live",
            "describes": "End point for subscribing to LIVE data"
        },
        "additionalResourceInfo": {
            "value": "http://rbccps.org/resourceInfo/{id}"
            "describes": "Additional information about the device"
        },
        "resourceAPIInfo": {
            "value": "http://rbccps.org/resourceInfo/api",
            "describes": "Information on how to use various APIs (access, "
```
update, cat) associated with this resource

"data_schema": {
  "type": "object",
  "properties": {
    "dataSamplingInstant": {
      "type": "number",
      "description": "Sampling Time in EPOCH format",
      "units": "seconds",
      "permissions": "read",
      "accessModifier": "public"
    },
    "caseTemperature": {
      "type": "number",
      "description": "Temperature of the device casing",
      "units": "degreeCelsius",
      "permissions": "read",
      "accessModifier": "public"
    },
    "powerConsumption": {
      "type": "number",
      "description": "Power consumption of the device",
      "units": "watts",
      "permissions": "read",
      "accessModifier": "public"
    },
    "luxOutput": {
      "type": "number",
      "description": "lux output of LED measured at LED",
      "units": "lux",
      "permissions": "read",
      "accessModifier": "public"
    },
    "ambientLux": {
      "type": "number",
      "description": "lux value of ambient",
      "units": "lux",
      "permissions": "read",
      "accessModifier": "public"
    },
    "targetPowerState": {
      "type": "string",
      "enum": [
        "ON",
        "OFF"
      ],
      "units": "dimensionless",
      "description": "If set to ON, turns ON the device. If OFF turns OFF the device. Writeable parameter. Writeable only allowed for authorized apps".
  }
}
"permissions": "read-write",
"accessModifier": "protected",
],
"targetBrightnessLevel": {
  "type": "number",
  "description": "Number between 0 to 100 to indicate the percentage brightness level. Writeable only allowed for authorized apps",
  "units": "percent",
  "permissions": "read-write",
  "accessModifier": "protected"
},
"targetControlPolicy": {
  "enum": [
    "AUTO_TIMER",
    "AUTO_LUX",
    "MANUAL"
  ],
  "units": "dimensionless",
  "permissions": "read-write",
  "description": "Indicates which of the behaviours the device should implement. AUTO_TIMER is timer based, AUTO_LUX uses ambient light and MANUAL is controlled by app. Writeable only allowed for authorized apps",
  "accessModifier": "protected"
},
"targetAutoTimerParams": {
  "type": "object",
  "permissions": "read-write",
  "properties": {
    "targetOnTime": {
      "type": "number",
      "description": "Indicates time of day in seconds from 12 midnight when device turns ON in AUTO_TIMER. Writeable only allowed for authorized apps",
      "units": "seconds",
      "accessModifier": "protected"
    },
    "targetOffTime": {
      "type": "number",
      "description": "Indicates time of day in seconds from 12 midnight when device turns OFF in AUTO_TIMER. Writeable only allowed for authorized apps",
      "units": "seconds",
      "accessModifier": "protected"
    }
  }
},
"targetAutoLuxParams": {
  "type": "object",
  "permissions": "read-write",
  "properties": {
  
  }
}
"targetOnLux": {
  "type": "number",
  "description": "Indicates ambient lux when device turns ON in AUTO_LUX. Writeable only allowed for authorized apps",
  "units": "lux",
  "accessModifier": "protected"
},
"targetOffLux": {
  "type": "number",
  "description": "Indicates ambient lux when device turns OFF in AUTO_LUX. Writeable only allowed for authorized apps",
  "units": "lux",
  "accessModifier": "protected"
}

"additionalProperties": false

"serialization_from_device":{
  "format": "protocol-buffers",
  "schema_ref": {
    "type": "proto 2",
    "link": "https://github.com/rbccps-iisc/smart_cities_applications_streetlight/blob/develop/proto_stm/txmsg/sensed.proto"
  }
}

"serialization_to_device":{
  "format": "protocol-buffers",
  "schema_ref": {
    "type": "proto 2",
    "link": "https://github.com/rbccps-iisc/smart_cities_applications_streetlight/blob/develop/proto_stm/rxmsg/actuated.proto"
  }
}

Figure 8: Example catalogue schema for a smart streetlight

There are three portions to the above schema, highlighted by the three shades. The first (in grey shading) pertains to static information. The second part (light blue shaded) provides meta-information about:

● Observations made by the streetlight (caseTemperature, ambientLux, etc.)
○ For example, the streetlight measures “caseTemperature” which is a number and it describes the temperature of the light casing and its units are “degreeCelcius”.

● Control parameters for this device (targetPowerState, targetBrightnessLevel, etc.)
  ○ These parameters have permissions keyword set to “read-write”.
  ○ For example, the parameter “targetPowerState” can be used to switch ON (and OFF) the streetlight. This parameter is a dimensionless quantity and its accessModifier set to be “protected” thereby implying that only authorised users can modify this parameter.

● Configuration parameters (samplingRate, etc.)

The green-shaded part pertains to the device management information, e.g., in this case it refers to the data serialization format (protocol buffers) used by the device to send/receive information.

The structure for streetlight catalog item is defined by the base schema “generic_iotdevice_schema.json”\(^\text{14}\). This information is contained within the static information part of the item using the json fields, namely “refCatalogueSchema” and “refCatalogueSchemaRelease”. To enable format validation of an uploaded catalog item, it should mandatorily mention the name of reference schema files and the schema release numbers.

![Figure 9: Streetlight observation and control data](https://github.com/rbccps-iisc/smart cities schemas)

To highlight the validation features of the catalog, we present an example of the observation (up-arrow) and control (down arrow) data packets for the streetlight

\(^\text{14}\) https://github.com/rbccps-iisc/smart cities schemas
device in Figure 9. A key point is that, the observation schema which is stored in catalog (e.g., see the example streetlight item above), can be directly used to validate the incoming observation packets. The same applies to the control packets in the reverse direction.

From the above example, we also note that a field is provided to capture information about the data-serialization formats used by the device. In bandwidth constrained scenarios, e.g., LPWAN networks\(^\text{15}\), JSON is not an efficient format. It may be required to use some efficient serialization formats, e.g., protocol-buffers\(^\text{16,17}\), to send/receive information from the device. A key observation is that, using above information, one can easily orchestrate data flows that hide these complexities. That is, a user may continue to interact with the device (through middleware) using JSON whereas the middleware/gateways communicate the devices using their preferred serialization formats.

Another important field in the above example is the “ontologyRef” field included within “latitude” and “longitude” fields. This field provides a reference to a site where further explanation as to the meaning/semantics of the “key” can be found. For example, a website explicitly created to hold the ontology (dictionary) for that IoT device parameters in a standard ontology language (for example RDF\(^\text{18}\)).

We also provide an example of the catalog entry for power meter in Appendix.

*We recommend that an Indian City specific dictionary/ontology be created (called, for example, isco: Indian Smart Cities Ontology) and which contains semantic descriptions of things which are unique to Indian cities and hence not available in other existing ontologies. The dictionary as well as the catalogue should be ideally made available in all Indian languages.*

The catalogue can be accessed and searched via HTTP(S) commands and hence will be accessible from anywhere in the web.

*We advocate a resource catalogue be a part of every smart city middleware instance and it adheres to a standard schema as well as an API format for querying.*

### 3.2.3. Data Brokers

One of the key requirements for the IT system is the ability to develop new services and applications which consume the data produced by the various data sources. In addition, these applications could be developed by different vendors using diverse frameworks and languages like Java, Ruby, Python, etc. Thus, it is important to provide a generic

\(^{15}\) https://www.link-labs.com/blog/past-present-future-lpwan

\(^{16}\) https://developers.google.com/protocol-buffers/


\(^{18}\) https://www.w3.org/OWL/
mechanism to share data between the IoT devices and consuming applications. The architectural pattern of a message bus or a message broker is a powerful mechanism to support decoupling and asynchrony between the producers and consumers. Broker technology is well established and many good open source and commercial broker implementations are available. Examples include RabbitMQ, Apache Kafka, etc. These are designed to scale well and hence can be adopted for city scale implementations. The brokers support streaming data (i.e., continuous flow of information) and hence are suitable for IoT Sensors. Applications can connect via streaming APIs and this can be indicated in the associated meta-data for the device. There are a diverse set of application protocols which can be used to implement the APIs and include: AMQP, STOMP, XMPP, WebSockets and HTTP (streaming) itself. The APIs and the protocols to access the brokers can be indicated in the catalogue and hence this will allow future proofing the middleware including migration to new broker and application protocol technologies.

3.2.4. Data Stores

Resource endpoints which allow one to access a specific instance or subset of the resource data items, will be serviced by Data stores (or databases). For example, these endpoints can be used for requesting the last known state of a streetlight or the values of the illuminance level over the past week from a particular street light. These will be serviced by database queries in the back end. The type of database used to store the resource values will be indicated in the meta-data for the resource in the catalogue and hence the appropriate APIs can be used by applications to access the desired information. This mechanism will allow a diverse set of stores to be used – each tuned for the kind of data represented by the resource. Most databases like MongoDB, Cassandra, Elasticsearch, InfluxDB, Graphite etc. already offer REST APIs for interactions. New database technologies can also be easily adopted in the future.

3.2.5. Video Broker and Store

Video is one of the key media for Smart Cities. Currently, it is being viewed primarily for surveillance and traffic monitoring applications. However, as can be seen from Appendix A, it can provide valuable information for many other services. Hence it is important to ensure that video data is easily available for other consuming applications (besides surveillance). Video being a very bandwidth intensive resource, it needs specialised brokers (also called streaming engines) and video management systems (to store the video from a number of cameras). Video management systems offer the ability to manage the cameras as well as control the recording. Most cameras now are IP enabled (directly accessible via an IP network) and “speak” ONVIF\(^\text{19}\), which is a standard interface format to configure and manage IP cameras. In order to enable new use cases for the video feeds, it will be important to be able to distribute the video to new

\(^{19}\text{https://www.onvif.org/}\)
consumers (e.g., new video analytics applications or show live feeds of events in certain places to the citizens) and this might require video streaming engines. Many engines are now available as cloud services and come with support for REST and Streaming APIs.

3.2.6. Authentication, Authorization, Accounting, Auditing, Logging and Middleware Management

Authentication of human users, organizations, applications and devices needs be supported in the middleware. Applications and devices need to be linked to users or organizations. Well-developed standards and implementations exist for these functionalities and these can be easily adopted\(^{20}\) (certificates, password credentials, API keys, tokens, certificates, etc). Support for single sign-on, time based credentials, application authentication (via signatures), device authentication (via certificates) and authorization tokens should be incorporated in the middleware.

Authorization refers to the ability of the owner to control access to the middleware resources and is enforced by the policy engine. Standards based authorization techniques should be adopted\(^{21}\) Examples include: Granting access to resources in a certain locality, granting access to resources of a certain type, granting write access to certain resources, or granting permission to upload only certain amounts of data per day.

Accounting keeps track of the usage/access patterns of the resources by every user, application, device and can be used for later billing. Interfaces to UPI etc can be provided to allow for easy payment.

Auditing might become an important requirement for Smart City transactions and services. Technologies like Blockchain\(^{22}\) are emerging which enable distributed trust and could be explored in the future.

Logging is necessary for both system debugging as well as auditing/forensics. Middleware management will help to keep track of the status of the various services, APIs, etc and help to administer them. This will mainly be important for the implementers of the system.

3.2.7. Policy Enforcement and data integrity checks

The policy enforcement engine polices the usage of resources as per the authorization policy. Basic integrity checks on the data formats, access requests, etc. are done here. Appropriate error messages are conveyed to the requester. No semantic understanding of the data is assumed (or possible) as these will be domain specific.


\(^{21}\)https://en.wikipedia.org/wiki/XACML.

\(^{22}\)https://en.wikipedia.org/wiki/Blockchain
3.3. Data Middleware APIs

A baseline middleware should support APIs to manage entities, access live data, publish live data, access catalog and access historical data. In addition, more APIs can be added for additional services like getting GIS data, or accessing live video streams. In this document we restrict our discussions to the baseline APIs.

An entity is any software which interacts with the middleware (IoT device or an application). Each entity is associated with a human user. The user first needs to obtain an account with the administrator of the middleware. Subsequently, the user registers a number of entities which actually interact with the middleware. The user account creation is offline and might involve additional aspects of verification (much like getting cellphone connections). Upon account creation, the user receives an API key (user_api_key), which is then used for registration of all the user's entities (devices, applications, etc.) via the registration API. The entities are registered to do one or more of subscribe, publish or get access to historical data. Each entity gets its own API Key (entity_api_key), which it uses in all subsequent interactions with the middleware.

As an example, here is a set of APIs as a baseline.

<table>
<thead>
<tr>
<th>APIs / Methods</th>
<th>/register</th>
<th>/publish</th>
<th>/subscribe</th>
<th>/cat</th>
<th>/db</th>
</tr>
</thead>
<tbody>
<tr>
<td>POST</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>GET</td>
<td>YES</td>
<td>-</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>PUT</td>
<td>YES</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>DELETE</td>
<td>YES</td>
<td>-</td>
<td>YES</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 1: Example Baseline REST APIs

3.3.1. Registration APIs

The following tables briefly explains the APIs for registration to endpoint https://<middleware>/register.

<table>
<thead>
<tr>
<th>Method</th>
<th>Parameters</th>
<th>Body</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>POST</td>
<td>user_api_key, resourceId, validityEndTime</td>
<td>Catalog resource schema in JSON SCHEMA format</td>
<td>The entity (either a device or an application) registers with the middleware. It also supplies meta information about</td>
</tr>
</tbody>
</table>
itself in the JSON SCHEMA format as discussed earlier as well as other registration parameters (e.g. validityEndTime). In return an entity_api_key is returned

<table>
<thead>
<tr>
<th>Method</th>
<th>Parameters</th>
<th>Body</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET</td>
<td>user_api_key, resourceld</td>
<td>NULL</td>
<td>Get the entity_api_key and registered catalog schema and other parameters</td>
</tr>
<tr>
<td>PUT</td>
<td>user_api_key, resourceld, validityEndTime</td>
<td>Catalog resource schema</td>
<td>Modifications to the entities resource schema stored in the catalog as well as other registration parameters</td>
</tr>
<tr>
<td>DELETE</td>
<td>user_api_key, resourceld</td>
<td>NULL</td>
<td>delete the entity from the middleware</td>
</tr>
</tbody>
</table>

3.3.2. Publish API
Endpoint: https://<middleware>/publish

<table>
<thead>
<tr>
<th>Method</th>
<th>Parameters</th>
<th>Body</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Post</td>
<td>entity_api_key</td>
<td>JSON Document</td>
<td>JSON Document as per the JSON Schema to be published</td>
</tr>
</tbody>
</table>

3.3.3. Subscribe API
Endpoint: https://<middleware>/subscribe

<table>
<thead>
<tr>
<th>Methods</th>
<th>Parameters</th>
<th>Body</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Post</td>
<td>entity_api_key</td>
<td>JSON with list of resources to</td>
<td>Request to subscribe to a list of</td>
</tr>
<tr>
<td>Method</td>
<td>Parameters</td>
<td>Body</td>
<td>Remarks</td>
</tr>
<tr>
<td>--------</td>
<td>------------</td>
<td>------</td>
<td>---------</td>
</tr>
<tr>
<td>Get</td>
<td>entity_api_key, resourceId</td>
<td>NULL</td>
<td>Opens a streaming connection to data from the resource.</td>
</tr>
<tr>
<td>Delete</td>
<td>entity_api_key</td>
<td>JSON with list of resources to unsubscribe</td>
<td></td>
</tr>
</tbody>
</table>

3.3.4. Catalog API
Endpoint: https://<middleware>/cat

<table>
<thead>
<tr>
<th>Method</th>
<th>Parameters</th>
<th>Body</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get</td>
<td>entity_api_key, query string</td>
<td>NULL</td>
<td>Returns catalog entries that match the query string</td>
</tr>
</tbody>
</table>

3.3.5. Database API
Endpoint: https://<middleware>/db

<table>
<thead>
<tr>
<th>Method</th>
<th>Parameters</th>
<th>Body</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get</td>
<td>entity_api_key, query string</td>
<td>NULL</td>
<td>Returns entries from the database that match the query string.</td>
</tr>
</tbody>
</table>

4. Security aspects for IoT implementation for smart-cities
Many Smart City IoT devices, like video cameras, smart streetlights, gateways, etc. will be deployed in public places and hence will be vulnerable to tampering. Therefore, particular care must be taken to make it difficult to compromise them as well as limit the damage they will do in case they do get compromised.

4.1. IoT Device Security

4.1.1. Default Username and Passwords
Reports\(^{23}\) suggest that 15% of IoT device owners don't change the default passwords.

This was one of the main reasons behind DDoS attacks by botnets like Mirai\textsuperscript{24}. These issues could be handled by providing long and complex random usernames and passwords for each device. Alternatively, automated certificate based authentication should be considered.

4.1.2. Principle of Least Privilege (Privilege Separation and Dropping)

For any communication with outside world an IoT device must use an unprivileged user mode with least permissions. For example, a process will typically require administrator/root privileges to open a privileged port at startup. However, after completing all the privileged tasks, the process must drop its privileges. Also, it is a good practice to make sure that the programs are written in a way to support privilege separation (e.g., OpenSSH\textsuperscript{25}). After dropping to an unprivileged user mode, restricting a process to a chroot/jail is an very effective way to reduce the damage caused due to an attack.

4.1.3. Capabilities

Even with an unprivileged user, by default a vulnerable service/program can be prone to certain attacks. Hence, capabilities of a service/program must be restricted to be as low as possible. Several such libraries are available on various platforms. See for example: Pledge\textsuperscript{26}, Seccomp\textsuperscript{27}, and Capcicum\textsuperscript{28}. These capabilities libraries help in reducing the impact on successful attack.

4.1.4. Firewall and Intrusion Detection and Prevention Systems

It is well known that some of the IoT cameras have been used to launch DDoS attack. Good firewall configuration must be used to control and throttle network packets. For a firewall, whitelists must be prefered over blacklists. Application firewalls and intrusion detection and prevention systems like fail-2-ban/tripwire can also be used to resist brute-force attacks.

4.1.5. Full Disk Encryption

IoT devices which are accessible to public can be stolen or tampered. This requires good physical security. However, from a software point of view, full disk encryption could be one of the options to consider. The only issue on using full disk encryption is that there must be a way provided to enter password if the device reboots.

\textsuperscript{24}\url{https://www.csoonline.com/article/3126924/security/here-are-the-61-passwords-that-powered-the-mirai-iot-botnet.html}
\textsuperscript{25} Security measures in OpenSSH, \url{https://www.openbsd.org/papers/openssh-measures-asiabsdcon2007-slides.pdf}
\textsuperscript{26} Pledge manpage : \url{https://man.openbsd.org/pledge.2}
\textsuperscript{27} SE Cure COMP uting with filters : \url{https://www.kernel.org/doc/Documentation/prctl/seccomp_filter.txt}
\textsuperscript{28} Capsicum: practical capabilities for UNIX : \url{https://www.cl.cam.ac.uk/research/security/capsicum/}
4.1.6. Certificates, PUFs, TPM and Physical Security

Use of security certificates and mutual authentication should be the preferred way for encrypted communication. Short-lived certificates should be preferred over long-lived certificates, and a secure workflow to update certificates periodically must be planned. Certificate revocation list server is a must and should be up-to-date. Where feasible, PUFs (Physically unclonable functions) may be used as identification of the device. If costs permit, using a hardware trusted platform module (TPM) should be used to store certificates and Keys to prevent them from being stolen.

Physical security like tamper detection are a must for IoT devices exposed to public. Such devices must be placed in tamper-resistant boxes and sensors must be installed to send alarms to control center in case of tamper. In software, heartbeats can be used to check if the device has been stolen/powered off by an attacker.

4.1.7. API security

If the IoT devices provide APIs, then while implementing them it is recommended to use a safe programming language or a restricted set of languages (like MISRA-c) wherever possible. Audits and checks using security scanners should be performed to check for possible attacks like buffer overflows, sql/command injection possibilities, denial of service, and cross-site scripting.

4.1.8. Secure Operating System

The operating system must provide a secure platform for IoT devices. The operating system must be chosen after careful evaluation of its security aspects and the tradeoff with performance. Also, capability based operating systems such as Google Fuchsia\(^\text{29}\) could be one of the options to consider in future.

4.1.9. Use of Cryptographically Secure Random Numbers

In general, pids, network ports, initial sequence numbers, etc. should be random whenever feasible. This renders some types of attack difficult in practice. Strong random numbers (i.e., cryptographically secure random numbers) or hardware based random numbers must be used wherever possible.

4.1.10. Use of Modern Constant Time Cryptography

As many of the IoT devices are accessible to public they are prone to side channel and timing attacks. Hence use of constant time cryptography (e.g., ChaCha20 + Poly1305) should be used. Also, old, outdated, and weak cryptographic algorithms must not be used.

\(^{29}\) Google Fuchsia: https://fuchsia.googlesource.com/
4.1.11. Software updates from trusted source

Software updates for IoT devices must be only allowed from trusted sources upon verifying the certificate of the remote host.

4.2. Middleware security

In this section, we discuss the securing of the data middleware. Security is an evolving field and it is of paramount importance to be abreast of the best practices.

![Security architecture of middleware](image)

**Figure 10**: Security architecture of middleware. The solid-arrows represent the data flow wrt. the user, and the dashed-arrows indicates data flow due to internal processes in the middleware. The black colored blocks in the middleware indicate the systems with which the users may interact with; the red colored blocks are related to security; and the green colored blocks are for miscellaneous purposes.

We illustrate various possible components for securing the data middleware using a specific example shown in figure 10. Various components and the security aspects of the middleware are:

4.2.1. Firewall cluster

The firewall is one of the critical component in the data middleware and is the only point of access to the middleware for authenticated users. The firewall will only allow authenticated and encrypted traffic, and will allow access to services offered by the middleware such as: (i) broker, (ii) persistence, (iii) catalog, etc.

To access the middleware services, the user has to provide a valid certificate to the firewall, upon which it will then forward the service request to the respective service. The firewall is data agnostic and will not interpret or modify the data flowing through it. The individual services would perform the authorization of the request before
providing the service.

The port/service forwarding function of the firewall will be limited for a list of allowed hosts and ports/services inside the middleware. The firewall will also limit the rate at which a user can send/receive data to prevent DoS attacks. As the firewall cluster is a set of nodes, more nodes may be added to improve availability, redundancy, and scalability. Several such firewall clusters may be added to provide further load balancing and redundancy. The users may connect to a particular firewall cluster based on a load balancing policy such as: round-robin (e.g., round-robin DNS).

4.2.2. Certificate Authority (CA)

Certificate authority provides certificates that can be used to authenticate with the firewall, and is provided to individual users and organizations. Users need to present their certificates to access middleware services. Even organizations may use their certificates to issue certificates for their users/devices/gateways.

For each user/organization, after API keys and certificates are generated, they are sent to the user/organization through email. Only the certificate id should be stored for future use and the actual keys and certificates should be deleted from the CA.

4.2.3. Security Policy Enforcer and Accounting

All the data flowing in and out should be monitored for accounting and compliance with security policy. The enforcer can be configured to take action if any of the entries attempts to violate the security policy.

4.2.4. NTP Server

In the middleware, activities such as issuing certificates, enforcing security policies, etc. require clocks of various systems to be in sync. Hence, a NTP server is required to distribute time for various systems inside middleware. The NTP server should operate in a network with least network congestion as possible.

4.2.5. DNS server

A DNS server (DNS with security extensions or DNSSEC) is required provide secure name resolutions. DNSSEC enables the DNS responses to be validated.

4.2.6. Authentication and Authorization Server

An AA server stores all the information related to authentication, authorization, and security policies. This information must be stored in a fast DB/directory as it is expected to be modified fewer times, but will be read often.
### 4.2.7. Security Attributes

Below are the security attributes of the middleware:

<table>
<thead>
<tr>
<th>Security aspect</th>
<th>Achieved by</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authentication</td>
<td>A valid certificate is required to connect to the middleware.</td>
</tr>
<tr>
<td></td>
<td>Also, each service in the middleware may require extra (2 factor) authentication using [username, password].</td>
</tr>
<tr>
<td>Authorization</td>
<td>For each service inside middleware, the AA server provides authorization information for the authenticated client.</td>
</tr>
<tr>
<td></td>
<td>For each service request, authorization is performed.</td>
</tr>
<tr>
<td>Accounting</td>
<td>All the traffic/logs for data from and to middleware is monitored by the “security policy enforcer”.</td>
</tr>
<tr>
<td></td>
<td>It can be configured to take actions on detection of attempts to violate security policies.</td>
</tr>
<tr>
<td>Confidentiality</td>
<td>A secure tunnel ensures that all data is confidential.</td>
</tr>
<tr>
<td>Integrity</td>
<td>Secure tunnel using PKI ensures that data can never be modified accidentally/intentionally.</td>
</tr>
<tr>
<td>Availability</td>
<td>The firewall cluster ensures load balancing and availability.</td>
</tr>
<tr>
<td></td>
<td>Security services may also be federated to provide high availability and</td>
</tr>
</tbody>
</table>
4.2.8. Example Implementations for Security in the Middleware

From the abstract middleware architecture (figure 10), two specific implementation architectures of the middleware are shown as examples in figures 11 and 12.

**Figure 11:** An example realization of the architecture given in figure 2. The solid-arrows represent the data flow wrt. the user, and the dashed-arrows indicates data flow due to internal processes in the middleware. The black colored blocks in the middleware indicate the systems with which the users may interact with; the red colored blocks are related to security; and the green colored blocks are for miscellaneous purposes.
Figure 12: Another example realization of the architecture given in figure 2. The solid-arrows represent the data flow wrt. the user, and the dashed-arrows indicates data flow due to internal processes in the middleware. The black colored blocks in the middleware indicate the systems with which the users may interact with; the red colored blocks are related to security; and the green colored blocks are for miscellaneous purposes.

4.2.9. Security Process Flow

The three main processes in accessing data from middleware are given below:

1. Creating accounts for individual users: The Smart City administrator will create user accounts in the middleware after gathering the required information from the user and will issue user-credential and user-api-key (figure 13).
2. **Creating accounts for user’s devices and applications:** The user would then want to deploy sensors to contribute data to middleware or applications to consume data from the middleware. The user will use his/her user-api-keys to request the API keys for user’s devices and applications (figure 14). The specific services for these devices and applications are also recorded by the middleware for later authorization checks. These device/application keys and certificates will be installed by the user in their respective device/applications.

3. **Accessing services of the middleware:** The user’s devices and applications will then register (onboard) themselves with the middleware. After this, they will have to present their API key to access the services of middleware (figure 15),
and will be granted access as per their authorization.

Figure 15: Process for requesting service with the middleware
5. A brief survey of a few IoT platforms

A Smart City platform is made up of multiple systems communicating with each other with authentication, authorisation and accounting of every message sent by devices; which can scale up to billions in the scope of a city.

An article by MachNations provides a classification of 5 types of IoT platforms:

![Figure 16: Five Platforms as reported in MachNations](image)

**An Application Enablement Platform (AEP)** is a technology-centric offering optimized to deliver a best-of-breed, industry-agnostic, extensible middleware core for building a set of interconnected or independent IoT solutions for customers. An AEP vendor relies on a flexible deployment model; a comprehensive set of device and enterprise-backend connector SDKs and APIs; and a set of well-documented developer resources. AEP vendors assemble a network of application development, system integrator and service provider partners that build custom IoT applications on the platform for customers. Enterprises realize that a well-built IoT application enablement platform (AEP) saves significant development time and money in the creation and operation of an IoT solution. Azure IoT Hub, AWS IoT, Ayla Networks are examples of AEP.

**A Platform-Enabled Solution (PES)** is a solution-centric offering optimized to reduce the time to deliver a fully-enabled, end-to-end, vertical IoT solution to customers. A PES vendor relies on a product-plus-services model; an application development lifecycle managed by in-house and partner resources; and either a rapid application builder or template-based framework. PES vendors use a direct or partner-led sales model and engage a core set of systems integrators to create validated, rapid and secure industry- or application-centric IoT solutions for customers. Philips CityTouch, a connected outdoor lighting offering from Philips, is an example of a PES.

**A Connected Device Platform (CDP)** is an offering optimized to provide complete carrier-grade lifecycle management of the connectivity element of an IoT device primarily for mobile network operators (MNOs) and mobile virtual network operators (MVNOs). A CDP vendor relies on custom carrier-managed implementations and a deep set of integration points with OSS/BSS for provisioning, billing, alerting and other
operational purposes. CDP vendors use a direct sales model to engage with carriers. Jasper Control Center is an example of a CDP.

A **Device Management Platform (DMP)** is an offering that provides device lifecycle management functionality associated with the deployment and management of IoT assets. Typical IoT assets include IoT gateways, retrofitted and new industrial equipment, and Linux-based Arduino-like devices. Typical DMP functionality includes firmware upgrades, security patching, alerting and reporting about specific metrics associated with IoT assets. Resin.io is an example of a DMP.

An **Analytics Platform (AP)** is an offering providing sophisticated data federation, statistical modeling and reporting tools to allow users to draw insights from ingested data. Usually, AP capabilities lay northbound of the data ingestion and processing functionality provided by other IoT platforms like AEPs and PESs. In essence, APs offer advanced analytics capabilities drawing using machine learning and artificial intelligence algorithms. SAP HANA is an example of an AP.

According to the comparison and based on the features and requirements, **Application Enablement Platform (AEP)** is the type of platform a Smart City would essentially require.

The features that are important for a Smart City IoT middleware platform are:

- Device management
- Data distribution
- Security
- Data Analytics and Storage
- Protocol Support

**Device management**

The lifecycle of the device is important for the city to know the operational status. There should be a system that allows participants to add a new device, update a device and remove a device from the platform. The system should maintain a list called as the “Catalog” containing details about the devices that are contracted to the city, which can be used by consumers to request for data. **[Software systems involved: Dashboard (Onboarding, Lifecycle management), Catalog]**

**Data distribution**

Distribution of data between a producer and multiple consumers is the most important system for a Smart City. This system should allow bi-directional message flows (data channel and control channel). For the scale of the city, we need to make sure that the distribution system (a.k.a. message broker) can handle telemetry messages swiftly. **[Software systems involved: Data Broker]**
Security

Compared to any other IoT solutions, for a Smart City the requirement of network security and information/data security is crucial. Both inbound and outbound messages from the city platform must be secured with strong encryption mechanisms to avoid eavesdropping. Also, the devices/applications should be authenticated, authorised to publish/subscribe data.

[Software systems involved: Security System (Authentication, Authorisation, Accounting)]

Data Analytics and Storage

On a city scale, we always have the big-data problem. The analytics system must be capable of providing Real-time analytics, batch-analytics, and predictive analytics. The system should have provisions to store analyzed and RAW data, and allow authenticated users to access it.

[Software systems involved: Analytics System (ETL and Rules Engine), Persistence]

Protocol Support

The platform must also support multiple protocols; to widen the reachability and adaptability with different manufacturers, so that the information can be pushed into the system.

IoT Middleware Platforms

There are around 360 IoT middleware platforms, which is an active M&A area. In this section we will discuss about few of them as per our requirements for a Smart City. We obviously cannot cover the whole gamut of these here - but pick a few to give a flavor of what is out there. Detailed reports can be purchased by the original authors of such surveys.

5.1. Contineo IoT PaaS

We start with a brief description of platform developed by an Indian startup that is offering an IoT platform as a service\textsuperscript{30}. There are not much details on the platform - but it seems to be built on several open source technologies like RabbitMQ, Hadoop etc and provides multi-tenanted support as well as REST API interfaces. There is also support for MQTT and it allows connection of LoRa, Zigbee end nodes. The platform has application enablement feature, dashboard, device management, and support for analytics.

\textsuperscript{30} https://www.contineonx.com/
5.2. Azure IoT Hub

The architecture of Azure IoT hub is as shown in figure 17.

![Figure 17: Azure IoT Hub](https://docs.microsoft.com/en-us/azure/iot-hub/iot-hub-what-is-iot-hub)

IoT Hub is a product by Microsoft that offers solution to connect and manage IoT devices. It provides a bi-directional communication between the Cloud and the Device with two end-points known as D2C (Device to Cloud) and C2D (Cloud to Device) per device. The device publishes sensed data at D2C end-point and accepts commands at C2D end-point. On the protocol front, Azure IoT Hub supports AMQP 1.0, HTTP 1.1 and MQTT 3.1.1.

IoT Hub maintains an identity registry for enabling and disabling a device. The registry also provides information about connection status, last activity time etc. With device identity management, we can create, retrieve, update and delete devices from IoT Hub. On the security front, IoT Hub uses TLS for secure network communication with X.509 certificates. For device authentication and access control of a device, IoT Hub uses a SAS (Shared Access Signature) token, which is verified against the device in the registry.

For data analytics, IoT Hub uses the services of Microsoft Azure Stream Analytics and Machine Learning solutions for providing real-time analytics and predictive-analytics.\(^\text{31}\)

---

\(^\text{31}\) Overview of the Azure IoT Hub service
5.3. AWS IoT

The architecture of AWS IoT is as shown in the figure 18.

Figure 18: Amazon AWS IoT

AWS IoT is a solution by Amazon that enables IoT devices to send data to the cloud. It uses the publish-subscribe mechanism similar to that of MQTT, where the sensed information is published to topics on the message broker. On the protocol front, AWS IoT supports HTTP 1.1, Websockets and MQTT.

Similar to Azure IoT hub, AWS IoT also maintains a Registry that assigns a unique ID for a device and stores device attributes and metadata. AWS IoT introduces the concept of Device Shadows, which creates a persistent virtual version of the device in Cloud. Applications do not directly interact with the physical device, instead they communicate the control messages to the device shadows through REST APIs. Once the device is up, it synchronizes with its shadow and obtains the messages. AWS IoT provides authentication and authorisation based on X.509 certificates, IAM, Amazon Cognito identities and Federated identities. These are used to verify the policies and grants access to the devices/application. The network communication is secured with TLS (mutual authentication).

With AWS IoT Rules Engine, AWS services such as Kinesis (real-time analytics), DynamoDB (Data storage) etc. can be utilized\(^{32}\).

---

\(^{32}\) Overview of AWS IoT service https://aws.amazon.com/iot-platform/how-it-works/
5.4. IBM IoT Platform

The architecture of IBM Watson IoT is as shown in the figure 19.

**Figure 19: IBM Watson IoT Platform**

Watson IoT platform provides IoT devices, application to interact with each other using publish-subscribe paradigms called topics. Every device in the platform has a model, type, attributes which describes the metadata and management characteristics. The information are captured while on-boarding the device and can be viewed, updated and queried. Each device and application will obtain a unique ID and API key to interact with the platform. Watson IoT platform provides APIs for Device Management with which the device can be rebooted, updated with firmware, flashed etc.

On the protocol front, the platform supports MQTT for devices, gateways, applications; HTTP based REST APIs for devices and applications. The communication network is secured with TLS and the device and applications are authenticated and authorised via API keys before accessing or publishing the data. The data is also encrypted to preserve information security. The platform uses unique topic structure to prevent IoT data leaking between silos.

Watson IoT platform provides a rules engine (Cloud rules and Edge rules) to perform real time analytics and trigger an actuation event for Application and Devices.\(^{33}\)

---

\(^{33}\) Overview of IBM Watson IoT Platform [https://console.bluemix.net/docs/services/IoT/index.html](https://console.bluemix.net/docs/services/IoT/index.html)
5.5. Oracle IoT Cloud Service

The architecture of Oracle IoT Cloud Service is as shown in the figure 20.

![Oracle IoT Cloud Service](image)

**Figure 20: ORACLE IoT Cloud Service**

Oracle Cloud service is an IoT cloud platform which provides a rich set of REST base tools that allows organisations to onboard their device with their set of resources, properties and data models. These devices will be provided with an access token while onboarding which they use for authentication. The data from the devices can be pushed to the Cloud service via HTTP REST or MQTT. For publishing and subscribing data from the service, a valid access token per session must be generated and presented along with HTTP basic authentication or OAuth. The data published is persisted within the Cloud for archival.

To subscribe for data, the application must present the endpoint in which it runs a web-service-listener. On subscription, the data is posted via HTTP POST by the Cloud Service to the endpoint specified during the subscription. The communication network is secured via SSL/TLS. The Cloud platform provides its Oracle Analytics Cloud services for performing real-time analytics, which can help to trigger alarms34.

---

34 Overview of ORACLE IoT Cloud Service
5.6. Google Cloud IoT Core

The architecture of Google Cloud IoT Core is as shown in the figure 21.

![Google Cloud IoT Core Diagram](image)

**Figure 21**: Google Cloud IoT Core

Google recently (May 2017) launched a service called Cloud IoT Core; a platform that allows globally distributed devices to connect and publish data to the cloud. Devices can publish data using the Cloud Pub/Sub, a messaging system similar to the lines of publish-subscribe paradigm, built around MQTT which is secured with TLS.

Cloud IoT Core acts as the platform that brings the edge devices to use integrated GCP (Google Cloud Platform) services and applications seamlessly. The data published is persisted using Cloud Storage and analytics is done through Cloud BigQuery. Stream and batch processing services such as Cloud Dataflow can be used too.

Apart from the above discussed platforms, there are many other enterprise software platforms developed by Bosch, C3, Thingworx, ConnectedThings, etc., which are used by city councils to build some solutions for the cities. For e.g., ConnectedThings is used by Italian cities like Nice, Bologna, Bordeaux, Strasbourg to build a Smart Public Transport System, Smart Airports, Shopping Centers. But the main question is how adaptable, scalable and functional are the available solutions to the overall idea Smart City Middleware.

Apart from the ones discussed above, there are several sources of IoT platforms:

---

35 Overview of Google IoT Core https://cloud.google.com/iot-core/
37 C3 IoT https://c3iot.com/products/c3-iot-platform/
38 The ThingWorx IoT Technology Platform https://www.thingworx.com/platforms/
39 ConnectedThings https://www.connectthings.com/
A. Carrier Platforms which are used to manage cellular networks such as (WebNMS for Zoho, Universal IoT Platforms from HPE), have adopted open source components to rapidly add IoT capabilities.

B. Open source initiatives are making a major inroad into this space and much of the underlying plumbing of all major IoT Platforms use open source components. Some notable open source platforms are the Kaa Project and Things.io. Open Connectivity Foundation (OCF) has sponsored the Iotivity project at the Linux Foundation. OneloTA.org focuses on automating the process of quickly adding new devices to a network, regardless of location and Eclipse has many projects ranging across multiple domains.

C. Many commercial offerings like Impact/Nokia and Thinger.io are built up from these but offer carrier grade management.

D. Another aspect of differentiation is objective. Some are focused on simple device management (Core need for SmartCity) like Resin.io some are full blown application development and analytics platform(SAP/HANA) and geared to developers( IBM Bluemix) and some are geared to end users allowing visual drag and drop application development (SmartThings)

We can clearly see that the solutions available are creating silos where the data/information access is strictly restricted for entities outside the scope of the organisation that is producing the data. We found good candidates which have rich features such as Analytics, Storage and Security. Data Distribution beyond the scope of the Owner of the device and Cataloging are features that enables a rich Smart City platform; which we found missing is the platforms available in the market. Also, having an Open Smart City Middleware Platform will enable innovations and rich applications to be built around the data.

5.7. BIG IoT

An European Union sponsored project called BIGIoT\(^{40}\) comes the closest to perspective presented in this report. This is a recently started project which aims to build an IoT market place and ecosystem where different IoT platforms can offer their IoT data. The consumers of this IoT data are Applications and Services. Services consume data and offer “refined data” and other services in the marketplace. Applications only consume data. The marketplace is similar to the data middleware discussed earlier - except for the broker and database services. The marketplace allows registration of resources, their discovery and purchase/payment. Actual data is served by the platforms or services, through client libraries\(^{41}\). The platform and libraries are currently only available to registered users and is currently under development and testing.

\(^{40}\) http://big-iot.eu/

6. Conclusions

“APIs Rule the World. GUI is no longer the king” -- Unknown source.

The current Smart City implementations are focused on delivering a command and control center, with a significant visualization component. While this is certainly an important piece of the smart city solution, it is not the most significant piece. The reason is that the complexity associated with the onrush of data will be too much to handle by humans in a direct way. Powerful data analytics will be needed to digest this data and aid human decision makers. This implies enabling machine access to data and hence APIs are the key. It will be great to have unified APIs across all the smart city platforms in India (even the world if possible). This will enable the same applications to work seamlessly across all Indian cities. This will require evolving a consensus on the APIs and data schemas, which is easier said than done. Regardless, it will be important to drive this effort via reference implementations to ensure that robust solutions emerge.

Another aspect we looked into was identifying India appropriate technologies. Digital technologies are fundamentally universal and applicable/usable by all humans. What will be India-Appropriate (in fact city-appropriate) will be customizations of this technology via local language support, specific data schemas for unique devices, specific city information models etc.

Suggestions for the future

Since there is not much experience with Smart City solutions in India, we recommend the setting up of sand-boxes/living labs/test-beds in every city to enable experimentation, learning and teaching of smart city concepts in these spaces. These can be co-located with a local university and involve local MSMEs and other companies, in collaboration with city administration. The smart city test bed will enable exploration of new concepts by researchers, demonstration of new technologies by entrepreneurs, teaching of smart city concepts to city employees and students and spur the development of local solutions to local problems. This will also help evolve unified APIs and data schemas for the smart city middleware. Every smart city can be encouraged to set aside a small budget for the smart city test bed, with additional support from state and central R&D funds. We expect about INR two Crores over a two-year period to be a good starting amount for a small test bed (this is 0.4% of the budget allocated by the centre for each Smart City).

Interoperability and compliance test centres and frameworks for Smart City solutions should be set up. Extensible, open source technologies and standards should be adopted/developed wherever possible to enable low cost/sustainable/vendor neutral solutions. Data models for all data sources (IoT devices, existing databases, applications etc.) needs to be developed. This will be essential for decoupling devices from applications and allow for devices from multiple vendors to interoperate. This activity
can be driven by one of the Indian standards bodies.

Indian city specific information model needs to be developed. This will be very useful for creating smart/intelligent applications of the future. This can be driven by one of the Indian standards bodies.

City, region & domain specific guidelines/standards in practically every sphere are needed. For example, smart street lighting standards in terms of minimum lux for safe city and hence the pole heights, LED lumens, installation guidelines, backup procedures etc. are needed for smart street lighting. Similar standards are required for surveillance (camera resolutions, deployment guidelines etc.) and practically every other application area. Indian Standards bodies should take these up one at a time to develop a thorough standard and reference implementation for each domain.

**Gaps in IoT technologies for Smart Cities for future R&D**

Low cost smart IoT devices for many city applications, customised to each city’s requirements, along with open data models are needed. For example, multi-color LEDs for North India, to effectively deal with fog in winter months.

Auditing of transactions will be important and in this regard, Blockchain technologies for distributed transaction audits need to be explored for their use in Smart City services.

Cameras are a big investment in most cities and we ought to be able to do more with them. Smart analytics for video will be essential to enable effective use of cameras and support many new applications. Distributed/edge analytics, close to the camera need to be explored, especially for video.
7. Appendix A: Analysis of Indian Smart City Components

In order to understand the various sensor devices which could be useful for smart city applications, we look at a specific application domain: intelligent transportation systems. Within this domain, there are a number of components related to managing transit centers, managing parking, traffic etc. We picked up about 24 components related to this domain and analyzed what sensors could be used to support them.

As an example, consider the component of Transit Operations Systems from the perspective of maintenance and tracking. One of key aspect will be know how many people and vehicles use the system per day and at what times. Further it will be useful to know which modes of transport people arrive in and depart (for e.g. bus vs. metros or which buses etc.). Clearly there is a need to track the number people and their movement patterns (without violating their privacy), the number of vehicles, their entry/exit patterns, their occupancy levels, etc. One can use one or more of pressure pads, connected toll gates (if smart cards are used), or cameras (with appropriate video analytics) for people counting and movement tracking. One could use IDs on the vehicles (like RFID or NFC based IDs), or use cameras (with ANPR analytics), to track vehicle movements within the center. Additionally, smart phone app based analytics could also potentially be done (based on wifi tracking etc). In addition, for effective management, one would also need to be able to communicate to the passengers, vehicle operators, as well as transit staff (via displays, megaphones, smart phone messages etc). Thus there is a diversity of technologies that are possible - each will have their pros/cons and need to be tried out in specific conditions.

The following table lists the various sensors/actuators needed to support the different components for the ITS.

<table>
<thead>
<tr>
<th>Smart City Component42</th>
<th>Devices</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transit Operations Systems (Maintenance and tracking)</td>
<td>Crowd sensors (cameras/phones/pressurepads), Bus ID and Location sensor (cameras/rfid/gps), Bus Bay Sensor (magnetometer/camera), Signage Actuators, Various other Asset Sensors (doors/lights/humidity/temperature)</td>
</tr>
<tr>
<td>Smart parking system</td>
<td>Parking sensors (Magnetometers/Cameras)</td>
</tr>
<tr>
<td>Area based traffic control</td>
<td>Traffic density sensors (camera/laser/magnetometers/cell tower records/phones), Signal sensors (LDR), Signal actuators (Light Controllers), Crowd sensors,</td>
</tr>
<tr>
<td>GPS tracking and optimisation of routes of garbage trucks</td>
<td>GPS sensors, other locationing sensors (UWB, WiFi, LoRa, CellTower Triangulation)</td>
</tr>
<tr>
<td>LED Streetlight lighting</td>
<td>LDR sensor, LED Control Actuator, Power Supply Sensor</td>
</tr>
</tbody>
</table>

42 Smart City Components, Arvind Tiwary, https://drive.google.com/file/d/0B12_7hDBCLoFM0Jid09Ccmdnc0U/view?ts=5729775e
| Traffic analysis or roads and video surveillance inside bus using CCTV surveillance | Cameras, crowd sensing |
| Fleet management system | Vehicle ID and Locationing Sensors (GPS, and other radio triangulation based sensors, RFID/BLE beacon based sensors) |
| Automatic fare collection system (transport) | Smart card readers and actuators/ smart phone (NFC) based devices/Biometric sensors |
| Pedestrian Infra | Camera sensors |
| Smart Bus stops | Live signages (LED Display actuators), Crowd sensors (Camera/pressure pads), Bus ID sensor (BLE/WIFI/Camera), Microphone/Speakers |
| City buses | GPS or other locationing sensors, Camera inside facing and outside facing, Driver/Bus/Operator ID, Capacity sensors, Empty seat sensors, Bus Health Sensors (Engine/Fuel/battery/tyre etc sensors), Energy harvestors (solar roof/panels) |
| In-bus information system and wifi | Speakers, Access Point,WAN connectivity, system health sensors |
| Private bus aggregator | Bus capacity/empty seat sensor, GPS & location sensor |
| Intelligent road asset management | Road and sidewalk quality sensor (crowd sourced/accelerometer/camera), Vehicle speed sensors |
| Bicycle pod with PIS | PoD lock sensors, Cycle presence sensor, Cycle ID sensor, User ID sensor |
| Smart Paving | Pressure based harvester, Solar Panels (as shades on pavements and double has harvesters) |
| Pedestrian and bicycle activated signals | Pedestrian/Bicycle Sensors (Cameras) |
| Junction improvement | Traffic sensor (camera/magnetometer/laser) |
| Paratransit facility | Same as for “Transit Operations System” |
| Bus Terminals | Same as for “Transit Operations System” |
| Bus Bays | Same as for “Transit Operations System” |
| Cycle Sharing | Cycle ID/Biometric Sensor/Cycle Location (GPS/Beacon) |
| Parking management | Parking sensors (camera/magnetometer), payment interfaces (phone) |
| E-Rickshaw | Solar based Charging stations with sensors for energy levels, GPS/Location sensors, Battery level sensor, |

We notice that cameras are a powerful sensor and could potentially be used to provide lots of information (besides being used for surveillance).
We give an example for a schema entry for a power meter below.

```json
{
    "refCatalogueSchema": "electric_meter_schema.json",
    "id": "rbccpsEnergy.EM_D0025860",
    "resourceType": "energymeter",
    "refCatalogueSchemaRelease": "0.1.0",
    "tags": [
        "electric meter",
        "Energy",
        "HVAC"
    ],
    "loadtype": "AC",
    "latitude": {
        "value": 13.0143335,
        "ontologyRef": "http://www.w3.org/2003/01/geo/wgs84_pos#"
    },
    "longitude": {
        "value": 77.5678424,
        "ontologyRef": "http://www.w3.org/2003/01/geo/wgs84_pos#"
    },
    "owner": {
        "name": "IISC",
        "website": "http://www.iisc.ac.in"
    },
    "provider": {
        "name": "Robert Bosch Centre for Cyber Physical Systems, IISc",
        "website": "http://rbccps.org"
    },
    "geoLocation": {
        "indoor_location": {
            "building_name": "Entrepreneurship Center",
            "floor": "3rd Flr",
            "room": "UPS Room",
            "address": "RBCCPS, IISC, Bangalore, 560012"
        }
    },
    "connection_topology": {
        "is_site_meter": false,
        "site_topology": "http://rbccps.org/resourceInfo/building/topology"
    },
    "accessMechanism": {
        "requestAccessSite": {
            "describes": "URI for getting permissions to access the device",
            "value": "http://rbccps.org/middleware/requestAccess"
        },
        "accessEndPoint": {
            "value": "https://rbccps.org/middleware/api/{api_ver}/db",
            "describes": "End point to access the archived values (database"
        }
    }
}
```
"access endpoint": "mqtt://rbccps.org/subscription/live",
  "describes": "End point for subscribing to LIVE data"
},
"resourceAPIInfo": {
  "value": "http://rbccps.org/resourceInfo/api",
  "describes": "Information on how to use various APIs (access, update, cat) associated with this resource"
}
}
"data_schema": {
  "type": "object",
  "properties": {
    "dataSamplingInstant": {
      "type": "number",
      "description": "Sampling Time in EPOCH format",
      "units": "seconds",
      "permissions": "read",
      "accessModifier": "public"
    },
    "RPhaseVoltage": {
      "type": "number",
      "description": "",
      "units": "volts",
      "permissions": "read",
      "accessModifier": "public"
    },
    "YPhaseVoltage": {
      "type": "number",
      "description": "",
      "units": "volts",
      "permissions": "read",
      "accessModifier": "public"
    },
    "BPhaseVoltage": {
      "type": "number",
      "description": "",
      "units": "volts",
      "permissions": "read",
      "accessModifier": "public"
    },
    "RPhaseCurrent": {
      "type": "number",
      "description": "",
      "units": "amperes",
      "permissions": "read",
      "accessModifier": "public"
    },
    "YPhaseCurrent": {

"type": "number",
"description": "",
"units": "amperes",
"permissions": "read",
"accessModifier": "public"
},
"BPhaseCurrent": {
"type": "number",
"description": "",
"units": "amperes",
"permissions": "read",
"accessModifier": "public"
},
"RPhaseActivePower": {
"type": "number",
"description": "",
"units": "watts",
"permissions": "read",
"accessModifier": "public"
},
"RPhaseReactivePower": {
"type": "number",
"description": "",
"units": "volt-ampere reactive",
"permissions": "read",
"accessModifier": "public"
},
"RPhaseApparentPower": {
"type": "number",
"description": "",
"units": "volt-ampere",
"permissions": "read",
"accessModifier": "public"
},
"YPhaseActivePower": {
"type": "number",
"description": "",
"units": "watts",
"permissions": "read",
"accessModifier": "public"
},
"YPhaseReactivePower": {
"type": "number",
"description": "",
"units": "volt-ampere reactive",
"permissions": "read",
"accessModifier": "public"
},
"YPhaseApparentPower": {
"type": "number",
"description": "",
"units": "volt-ampere",
"permissions": "read",
"accessModifier": "public"
}
"units": "voltage-ampere",
  "permissions": "read",
  "accessModifier": "public"
},

"BPhaseActivePower": {
  "type": "number",
  "description": "",
  "units": "watts",
  "permissions": "read",
  "accessModifier": "public"
},

"BPhaseReactivePower": {
  "type": "number",
  "description": "",
  "units": "voltage-ampere reactive",
  "permissions": "read",
  "accessModifier": "public"
},

"BPhaseApparentPower": {
  "type": "number",
  "description": "",
  "units": "voltage-ampere",
  "permissions": "read",
  "accessModifier": "public"
},

"RPhasePowerFactor": {
  "type": "number",
  "description": "",
  "units": "dimensionless",
  "permissions": "read",
  "accessModifier": "public"
},

"YPhasePowerFactor": {
  "type": "number",
  "description": "",
  "units": "dimensionless",
  "permissions": "read",
  "accessModifier": "public"
},

"BPhasePowerFactor": {
  "type": "number",
  "description": "",
  "units": "dimensionless",
  "permissions": "read",
  "accessModifier": "public"
},

"EnergyActive": {
  "type": "number",
  "description": "",
  "units": "kilowatt-hour",
  "permissions": "read",
  "accessModifier": "public"}
There are three portions to the above schema, highlighted by the three shades. The first (in grey shading) pertains to static information. The second part is the observation sub-schema and provides meta-information about the observations made by the electric meter, e.g., units, type etc. The last part (green-shaded) pertains to the device management specific information, e.g., in this case it refers to the data serialization format used by the device to send the observations to the middleware.
RFP Guidelines

9.1. Background

This is a draft report proposing considerations for RFP to be issued under the Smart City Mission (http://smartcities.gov.in) and focuses on unusual aspects from the technology aspect (IoT). This is prepared with contributions from many professionals in individual capacity under the IoT for Smart City Forum (IoT4SCTF) as well as published documents from many organizations and various guidelines on tenders and procurement. We have also relied on the draft e-Governance standards and guidelines as they cover hardware, Software and System Integration.

The report is aimed at MoUD and City Commissioners or CEO of the SPV formed to execute Smart City Mission.

9.2. Considerations

The proposals under Round 1 of the Smart City challenge cover a diverse range (81 solution components listed by MoUD) but can be conceptualized in several clusters with Operations & Control Centre (OCC), Transportation, Waste, Water and Safety as the more popular themes and a focus on inclusiveness and information sharing with residents.

From a technology and IoT/M2M perspective there are some special considerations for Smart City Mission.

9.2.1. Reference Implementations

There are no fully implemented and proven models of all the applications in scope. Some cities abroad have one or two applications (solution components) fully implemented and some have multiple scenarios implemented on a pilot for a small area. This has implications. It may be difficult to specify requirements when a great deal is imagined without concrete reference solutions to use as a base. It may also be difficult to get many vendors to select from in the tenders; reducing competition and creating concentration of same vendors implementing many cities. There also may be teething issues in the technology especially with Indian conditions that may have not yet been ironed out.

9.2.2. Rapidly changing Technology

Common methods like BOT or BOOT or Design-Build-Finance-Operate (DBFO) may assume too much in terms of stability of requirement, proven solutions and stable technology. For example, the last mile for sensors to connect up is currently designed with WiFi or 2G/3G but has proven worldwide to be an overkill. Alternatives for Low

---

43 http://www.deity.gov.in/content/model-rfps-e-governance-project
44 http://smartcities.gov.in/writereaddata/Smart_Solutions_Components.pdf
Power Wide Area Network (LPWAN) like LoRa and HaLow and NB IoT are being piloted and are projected to reduce cost of connectivity by at least 10X. A simplistic back of the envelope calculation shows that for a billion devices like that required for Mumbai the power requirement for WiFi/3G will be around 170-200 MW. LPWAN may reduce that by 500-1000 times to 170-400KW. This is significant and matters for ongoing operations.\textsuperscript{45}

We recommend cities should explore and use non licensed sub gigahertz spectrum and non-internet based communication technology esp IoT or capillary and mesh networks. We would encourage discussions with the IoT4SCTY Forum to dispel apprehension on this score.

\textbf{9.2.3. Buying Improving technology}

Since Smart City rollouts will take three to five years to be operational and last for seven to twelve years how do we take advantage of reducing cost and vastly improved quality due to new technologies? Bids for Solar power have followed a steep downward curve in India. Can we benefit from a similar curve?

\textbf{9.2.3.1. Price for Service}

One way is to define a service and ask for providers to quote a cost for that service to a SLA. However instead of a level 20 year costs as is common in solar power projects we recommend a formula that incentivizes the provider to rebase technology and offer reduced price over time. The provider has to actively become more productive with time. This is common in large IT or BPO deals in the private sector. This allows more realistic pricing and reduces mistakes of both types. Overly aggressive low price leading to provider having cash flow issues or too high a price as dramatic reductions become possible due to changes. Consider that price of oil has fallen by 50%. A formula for transportation services linked to oil prices would deliver benefits to the city while a fixed level cost quote would not deliver the full benefit.\textsuperscript{46}

\textbf{9.2.3.2. Phased additions}

It is almost certain that some new features will be needed in most solutions and some unanticipated changes may be required. Change control or a mechanism to review cost of changes and incorporating some changes is vital. We recommend that only minimum features be tendered in the first phase and only after few years add on features with suitable adaptation to alternatives be put to tender. This is similar to large IT projects being broken into many small sprints.

\textsuperscript{45} Spectrum required for very dense deployments is also an issue and the IoT/M2M community [TIE IoTForum (http://www.iotforindia.org/) for example] has requested for dedicated 10Mhz allocation in the sub gigahertz band. See http://www.iotforindia.org/wp-content/uploads/2014/10/WPC-Spectrum-V3.pdf

\textsuperscript{46} Variable pricing has risks for the City as prices can go up. However, vendors are in business to make reasonable profits and will unlikely absorb large losses so the city needs to be realistic in locking prices.
9.3. Shared Infrastructure

Many experts and commentators have noted the siloed nature of Government contracting in the Smart City space and also the solutions provided by vendors. Traditionally each utility like traffic police or energy have done this in their own way and continue to dig up roads one after other to lay “their” pipes.

Incentives to reuse a common communication infrastructure may help better utilization and more uptime (reliability). It may also be necessary as skills to manage multiple assets may simply not be available. Cyber security and protection from cross border cyber-attacks require very high level of skills and maturity.

A popular emerging idea in smart city projects and being pushed by Singapore is the pivotal role of street light poles. These are ideal arteries of an urban environment. A smart light pole every 250-500 meters may provide the smarts at a much more effective and efficient manner. This is a public asset, is powered and safe. Equip these with vandal proof IoT scaffolding with sensors to measure pollution, noise, precipitation, wind, measure crowd, cameras to count vehicular traffic and surveillance. See the Array of Things project 47.

The micro level data on weather, pollution etc may be an overkill from traditional ways to measure and report 48 but as urban densities increase and Big Data approaches gain acceptance this may prove the killer application. Micro level near real time data may allow a quantum leap in understanding the emergence of surges of demand and how to manage. This would provide a smart infrastructure platform which can be ready very fast. This will also expand as the city expands. The smart light pole can use a mesh network (IoT or capillary network) to share data and provide a way for citizens to send data to the City open data portal 49.

Corporate campus, academic campus, factories, shopping centres as well as gated communities should also be able to deploy and feed into the open data portal using this approach.

We strongly recommend standardization work for the Indian Smart City mission around a SmartLightPole based platform for the city.

9.4. Security

Smart city should also be Safe City. Smart Infrastructure is amenable to cyber terrorist attack of a different kind especially cross border attacks. Smart city also uses devices and communication equipment like gateways and routers in public places. These may

---

47 Array of things http://arrayofthings.github.io. Every 15 seconds these sensors will gather information like temperature, humidity, light, carbon monoxide (CO), nitrogen dioxide (NO2), and vibration. Precipitation and wind measurements will be added later. A sound sensor will collect data on ambient volume. An infrared camera, pointed at nearby sidewalk and roads, collects surface temperature information. A wireless network can count the number of Bluetooth- or WiFi-enabled devices in the vicinity of the node, as a proxy for pedestrian traffic.

48 Pollution in a big city like Delhi is measured at 9 locations. For rigorous health management and mitigation we need at least one per SqKm that is around 50,000 sensors.

49 Bluetooth esp new low power long range Bluetooth connectivity may allow phones to connect as far as 30 meters away and send data.
be tampered with and be used in attacks. Guarding against these require a very high level of vigilance over and above what is used by most web or cloud based systems.

The MoUD has issued a Cyber Security Model Framework for Smart Cities. This provides best practice from the web and cloud based world and is based on existing paradigm of passive defence and fortification of exposed assets. We add some points specifically for the devices or sensors layer as these will lack power, storage and compute capacity to meet internet standards.

We recommend an additional overlaid offensive approach which requires an above normal involvement from communication operators and device manufacturers and system integrators.

We recommend a central cell under NCIIPC (National Critical Information Infrastructure Protection Centre) to develop methods to test Smart infrastructure (whitelist hackers) and arrange emergency response in situations of attack. Each City and the central cell should also use active offensive methods. The cyber domain should be proactively patrolled and anomalous activity or intrusion attempts to enter the Smart infrastructure be actively tested. It should be easy for citizens and administrators to report suspicious activity and initiate automated logging over the network. Network operators should proactively monitor excessive activity, intrusion attempts and compromised devices and be authorised to take preventive actions. They should be authorized and required to keep logs of activity over the network and be able to identify initiators of queries and messages. The Smart Infrastructure network should not be the default Internet and steps should be taken to use trusted, secure networks (UID Aadhar is an example).

It is important to delegate police authority and enable self-defence in cyber domain because attackers tend to be change location and methods quickly. Traditional methods of capturing proof and taking action will be too little and too late. Self-defence rights should include disabling attackers or suspected attackers cyber control and command servers and isolating or denying access to the internet for such suspected attackers.

9.5. Open Data and Citizen services (Reimagined)

The full benefit of Smart City comes from empowering citizens to both get information and participate in solving as well as shaping future evolution. Technology can enable this empowerment but requires governance changes. From a ICT framework the most important need is open data. The range of solution components do not lend to a single standard at this point of time. They will evolve. However, the data must be shared not just with the utilities and administrators but with the citizens.

---


51 Item 25 of MoUD Security memorandum: Security Information and Event Management (SIEM) monitoring on all Smart City networks, devices and sensors to identify malicious traffic.

52 Item 12 and 20 of MoUD Security memorandum.
9.5.1. Benchmarking

We recommend capturing data in a format that allows sharing and future benchmarking as more experience accumulates. Thus four to five years later we expect a great deal of analysis of the data to showcase better and better ways to deliver services in terms of inclusion, effectiveness and efficiency. This needs standard definitions of data (household, facility) and collection to ensure metrics and KPI are correct and usable across cities. BIS is working on “ISO 37120:2014 Sustainable Development of Communities: Indicators for city services and quality of life “and cities should plan to provide data in these forms to enable benchmarking and data driven decision making. This can be a powerful learning across around 100 cities and lead to developing standards and best practices for Indian conditions.

9.5.2. Analytics + Big Data=Smart

Open data has led to surprising solutions as people from outside the incumbent hierarchy bring new perspectives and innovative approaches. This is a hard problem as many solution components have little digital data in a format that can be shared. We propose a bare bones approach to make that possible. We expect cities to launch challenges to encourage start-ups and others to analyse data in novel way and use big data and analytics to develop new insight and help with better predictive models to optimize resources and match demand and supply of urban services. There is a need to make data available at scale. This is also the philosophy of gov.in

The city digital platform/ fabric is based on open data.

9.6. Start-ups and MSME Inclusion

Many parts of smart city projects especially long running infrastructure will require a large system integrator. However, some work like training, support or acceptance testing may be better done by local SME. Similarly, proven technology may not be available or be just too expensive and it may be wiser to innovate and use newer technology. System Integrators will be loath to use unproven technology as typical tenders have very strong penalties for failure. Larger public policy may be better served with the city taking a bit more risk in a well disciplined and controlled manner and asking for PoC or pilots with start-ups providing newer more effective and efficient technology. System Integrators may need to be incentivized to use MSME and start-ups.

9.7. Citizen participation

India is rapidly urbanizing and growth of urban population stresses all aspects. From planning to providing capacity and matching demand to supply as well as monitoring breakdowns. It is important to re-imagine services with two aspects

---

53 ISO 37120:2014 Sustainable Development of Communities: Indicators for city services and quality of life is a base and BIS is enhancing this.
9.7.1. Crowdsourcing

Citizens can contribute data. They can buy and install devices like pollution and water meters on their own and also use smartphones. Smartphones are increasingly equipped with many high end sensors and GPS and photography and video capability and can be extremely effective in data capture.

We encourage cities to positively include such data and contribution in their process. Data governance approaches should identify such source and apply suitable checks. However, as a philosophy merrily because they are created by a non-controlled device they do not become useless. On the contrary engaged citizens may go to great lengths to calibrate and measure pollution, noise etc as it effects their health directly. It may be worthwhile to task the utility to investigate and reconcile discrepancy instead of ostrich-in-sand approach of ignoring external data.

9.7.2. Feedback Loop

In a digital India citizen aspirations are high. Citizens expect to voice their issues and receive response and shape policy and administrative action quickly. It is suggested that each project actively rethink digital engagement. Citizens should be able to report issues (bus stop which is no longer functional, water leaks, broken traffic lights) digitally (App, SMS, website, WhatsApp, Twitter, Facebook) and also see the resolution. Citizens should also be able to analyse statistics of issues and suggest changes like relocation of bus stops, changes in bus routes, etc.).

9.8. Interoperability, Standards and Reference Architecture

Smart Infrastructure as such is not new (SCADA has been around for decades) but the new pervasive deployment is new. Standards and best practices are evolving. A major constraint is that many smart infrastructure devices have to work in open public places under somewhat adverse environmental conditions and meet tight cost and power usage requirements. This renders many existing approaches from SmartFactory and SmartHomes somewhat unfit for purpose.

We suggest a pragmatic approach to standards as many internet standards or practices from developed economy may not be fit-for-purpose for dense, resource constrained Indian environment.

9.8.1. Interoperability

This is a big issue for technologists. It can be achieved by using standard components adhering to documented interfaces (API), data and connectivity as well as command and control. A reference architecture is also needed. The IoT4SCTF is in the process of developing one. This is a large undertaking and will take some time. We suggest RFP to be issued after considering the following ideas
- Competition for implementation by allowing different “wards” to be “managed” by different subcontractors. Can street lighting be done by different contractors and different components in different parts of the City.
- Ability to change components and migrate to newer technology in a staged manner
- Incentives for modernizing the application esp newer sensors and communication methods
- Incentives for sharing or re-using assets especially meters / sensors and communication infrastructure.

9.8.2. Device Management and Provisioning Standards

A key requirement for a central operations center to act on smart infrastructure is a standard way devices are provisioned and controlled. Example are TV Set top boxes or Cellular GSM world. This covers:

- Adding a new device and configuring it (Encryption, power usage and billing for communication)
- Shutting a device
- Upgrading a device

9.8.3. Operations (OCC) or Network Control Centre (NOC)

The Operations control center (OCC) or NoC should take care of the monitoring and managing networks of connected devices across categories and also seek to preempt degradations. NoC performs various maintenance activities like operations management, detecting fault, ticketing, performance evaluation. The more efficient the NoC the quicker it will be to solve the issues. The NoC should be able to support the deployment of applications in both hosted and cloud environments so that it can scale automatically to meet demand. At a minimum the NoC should cover the basic checks of managing devices to verify:

- If the device is up?
- Is it misbehaving?
- Data is in Red Zone?
- Can we shutdown/start/update?
- Can we bring the device down or disconnect from a node?
- Commands to the device to change a process (Water supply, Red light for VIP visit)
- Scheduling preventive maintenance

9.8.4. Common Components

There are multiple projects underway under Digital India, e-Governance and across different cities. Aadhar for authentication, Universal payment gateway and wallets
being launched by the new payment banks and other such should be part of converged requirements.

A vital common component for Smart City is spatial or map data and GIS. Cities are expanding fast. It is not unusual to find a road is no longer available or a Bus stop has shifted. Utilities need to have an accurate and up-to-date view on sensors, devices, connectivity hardware around the city. Regulatory approaches mandating the data ownership and process of updating as currently envisaged are simply unworkable.

It must be possible to have a quick (and dirty) update from crowd sourced inputs and publish with some delay (not a year later) an authoritative version.

10. RFP methods

There is a vast number of procurement methods as EPC (Engineering Procurement and Construction), BOO (Build Operate Own), BOT (Build Operate Transfer) and LROT (Lease Renovate Operate Transfer). The tender process may be single phase with a RFP or multi-phase with EOI followed by RFP as well as PoC or pilots as part of RFP phase. Multi-phase is used when requirement cannot be formulated and there is a lack of understanding on options for solutions. Where technology is not yet proven PoC or pilots should be planned.

10.1. RFP toolkit

The following aspects should be considered

**Process**

Tender can be single stage or multiple stage with an exploratory EOI and PoC or pilot

**Scoring**

Scoring can be on a single figure like lowest cost or a combination of commercial and technical variables like in QCBS

**Ownership**

Assets and IP can be always owned or ultimately transferred to the city (asset heavy) or the city consumes services and assets are provided by the provider (asset light)

**Eligibility**

Can bidder or sub-contractor be a consortium (SPV) or JV and LLP SME or Start-up. Do they need to have a presence in India and in the state?
10.1.1. Cheapest is not Best

Single stage fixed cost tender using lowest cost or L1 is the dominant method in practice. However, this has several well-known problems as discussed in this article in the Hindu.\textsuperscript{54}

A simple introduction to considerations in infrastructure projects (Smart City) may be found at http://toolkit.pppinindia.com/ports/module2-leapsfp-dotpp.php?links=fig9.

10.1.2. Quality cum Cost based selection (QCBS)

For more complex situations especially where the requirements or budget cannot be derived easily alternative scoring method which does not use a single cost metric should be used. Frequently the cost is related to quality. So we may want 90% uptime for network as a minimum eligibility. We may want to consider trade off in cost versus quality. So 95% or 99% uptimes costs may be considered. Similarly, we may want consider trade off to quicker implementation and cost. Rolling out to cover 75% of the population two years earlier at only 10% more cost may be better. This can be done thru Quality cum Cost based selection (QCBS). The world bank is a big user of this system and it has been used in India by infrastructure projects also. We recommend this strongly.

It is common to assign 30% weight to commercial parameters and 70% to non-commercial or technical parameters.

<table>
<thead>
<tr>
<th>Category</th>
<th>Feature</th>
<th>Weightage</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Commercial</td>
<td>CAPEX</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5 year OPEX</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Escalation</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>Technical</td>
<td>Uptime</td>
<td>70</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Coverage after 18 months</td>
<td>5</td>
<td></td>
</tr>
</tbody>
</table>

\begin{tabular}{|c|c|c|}
\hline
   & Uptime & Score \\
\hline
L1    & <5% over L1 & 25    \\
<10% over L1 & 20   \\
<15% over L1 & 10   \\
<20% over L1 & 5    \\
>20% over L1 & 1    \\
\hline
\end{tabular}

\begin{tabular}{|c|c|}
\hline
   & <90% Disqualified, Minimum Eligibility \\
\hline
<95 & 2    \\
<99 & 4    \\
>99 & 5    \\
\hline
\end{tabular}

Figure 22: Simple illustration of QCBS

10.2. Risk Reward

It is common in government tenders to transfer all the risk to the vendor. So road development vendors have had to absorb delays from government departments, green tribunal as well as much lower realized traffic. It may be safer from a bureaucratic point.

\textsuperscript{54}http://www.thehindubusinessline.com/opinion/procurement-challenge-in-public-services/article5892416.ece
of view to stick to this but the end users suffer due to very long delays and the cost to
the economy may be prohibitive.

The city should take a more proactive role and have incentives to make things happen.
So for example a minimum volume and period guarantee for a service-at-a-price
contract would make the vendor provide very good offers. The City can discover
different price points for different levels of guaranteed volumes and make a decision
based on realistic assessment of what can be done. This will also force different
government departments to coordinate far more actively then in the past. This is
anyway a prerequisite for the long term vision of Smart City Mission.
## Template for QCBS

<table>
<thead>
<tr>
<th>Commercial</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial Cost</td>
<td>10 (Equipment, SIM, License, CAPEX)</td>
</tr>
<tr>
<td>Training</td>
<td></td>
</tr>
<tr>
<td>Implementation</td>
<td></td>
</tr>
<tr>
<td>Cost till Full Rollout</td>
<td>5</td>
</tr>
<tr>
<td>OPEX for 5 years after full rollout</td>
<td>10 (Power, Connectivity, replacements)</td>
</tr>
<tr>
<td>Include maintenance, upgradess</td>
<td></td>
</tr>
<tr>
<td>Price Guarantee</td>
<td>5 (Escalation, Reduction Productivity improvement, Technology refresh)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Technical</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Functionality</td>
<td>15</td>
</tr>
<tr>
<td>NPR (Technology)</td>
<td>15</td>
</tr>
</tbody>
</table>

- Shared Infrastructure/Layered
- Connectivity
- Security
- Device Management
- High Availability / Scalability
- Peak Capacity / Annual Growth
- Interoperability
- Network Topology / Open Standards / Published APIs
- Connectivity
- Mesh / Sub-Gigabit
- Industry
- Language
- Disaster Recovery
- Cybersecurity
- Natural Disaster
- Operationalак
- Security Incident
- Time to Repair

<table>
<thead>
<tr>
<th>Implementation</th>
<th>20</th>
</tr>
</thead>
<tbody>
<tr>
<td>Months for 50% Coverage</td>
<td>1</td>
</tr>
<tr>
<td>Months for 75% Coverage</td>
<td>1</td>
</tr>
<tr>
<td>Months for Full Rollout</td>
<td>19</td>
</tr>
<tr>
<td>Methodology</td>
<td>1</td>
</tr>
<tr>
<td>Risk Management / Recovery Plan</td>
<td>1</td>
</tr>
<tr>
<td>System Integration Plan</td>
<td>1</td>
</tr>
<tr>
<td>Change Management</td>
<td>1</td>
</tr>
<tr>
<td>Citizen, Other Stakeholder Participation</td>
<td>1</td>
</tr>
<tr>
<td>Local Support</td>
<td>1.5</td>
</tr>
</tbody>
</table>

### Specific Reqs

- India
  - Power, Duct, Access/SSM
- City
  - Asset Inventory

<table>
<thead>
<tr>
<th>References</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>India projects</td>
<td></td>
</tr>
<tr>
<td>International projects</td>
<td></td>
</tr>
</tbody>
</table>

| Support | 5 |
| Technology refresh | 1 |
| Issue Management | 2 |
| Warranty & MRO | 1 |

<table>
<thead>
<tr>
<th>OSI/WEB Utilization</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Online</td>
<td>8</td>
</tr>
<tr>
<td>Offline</td>
<td>2</td>
</tr>
</tbody>
</table>

### Innovation |  |
11. **Annexure Open Data Portal**

The Smart City usage scenarios cover many diverse domains and technology with limited standards on communication technology, data standards or metrics. There are activities under way by BIS and ISO in this space.

As a base measure we recommend all solutions should place data in a meta data annotated format and in a machine readable format for future analysis. Big data and analytics techniques can be used on such data.

All the devices need to conform to some standard that is applicable in that particular domain and as an extra provide a publicly visible XML schema describing the data and definitions of the data used. *We should require all contractors to provide such a metadata XML and keep it up to date.* This needs some agreed data items like Contractor, City, Manufacture, Device Type ID, version ID and some methods are required to place such a schema in place.

### 11.1. Metrics

For benchmarking and coverage purposes we need to agree methods of counting Individuals, Household and facilities and space served. So a metric as cost / (SqKm*Household) may be a useful to compare and contrast projects and cities.

The following data set at a minimum level in the gov.in type portal for the city.

a. **Capability DB**: Device capability XML, JSON are good schemas to represent this data as it has wide support and we need to have the:
   i. Unique ID for device type
   ii. Manufacturer, Model and Edition
   iii. Contact information for technical support
   iv. Model ID (Device ID from manufacturer)
   v. Device category (Pollution, Water, Power, Camera)
   vi. Static or Mobile device
   vii. Channel Type (A device may report multiple types of data in different channels. Current power usage, 15 minute average, Daily usage )
   viii. For each channel What the device reports (units of measure, accuracy, sampling)
   ix. Definition of machine data pumped out by device across each channel
   x. Time stamp & Audit info

b. **Inventory DB**: Assets details. Capturing mainly on the inventory details
   i. Device ID (To recognize the device)
   ii. Capability ID (To identify the device manufacturer and capability)
   iii. Owner Contact details (For error and support communication)
   iv. Location type (Mobile, Static in House, In public place, In a facility)
   v. Geographic location
vi. Device identifiers for Utility or System integrator (RR meter for electric meters)

vii. Device Firmware, Software version

viii. Time stamp and audit info (basically who owns the device, updates, last used, last installed etc.)

c. **Measurement DB**: Raw data: The data which is not processed and in crude form.
   i. Device ID
   ii. Chanel Type
   iii. Measurement data
   iv. Audit data (Time stamp, Geolocation, gateway used)
12. Annexure Security for IoT or Devices layer in Smart City

12.1. Unique challenges for IoT Devices

Life of IoT Devices is in decades. This will outlast many security algorithms and techniques in place today. For example hashing using MD5 (which has been broken) has been replaced with RSA-2 but a energy meter installed a decade ago will continue with old hash algorithms. Implementing firmware or application software update is very complex and problematic as secure and error free update process substantially pushes up cost of the device and maintenance requirements. Keeping things simple is far better. Most modern browsers are fairly complex and have huge security flaws. It is not unusual to have daily security patches to be pushed out. Such large volume of updates is simply unmanageable in most current IoT networks.

IoT devices are manufactured in large volumes and may be installed in thousands. There may be generic configuration and common encryption keys and passwords used by implementers. IoT Devices may lack human interface and use embedded credentials to communicate in M2M usage scenarios. Static hard coded credentials can be reverse engineered and used to break in. IoT Devices may be shifted from one place to another, may be sold to a different user and may lack re configuration options. The IoT/M2M endpoints must be fingerprinted by means that do not require human interaction. Such identifiers include radio-frequency identification (RFID), shared secret, SIM, X.509 certificates, the MAC address of the endpoint, or some type of immutable hardware based root of trust. Auto configuration options should generate unique credentials but be manageable by automated remote management systems. IoT Devices may not be managed by IT departments in end user facility but by business or home users.

Many IoT Devices lack memory and compute capacity to run common security algorithms. Cryptographic suites such as Advanced Encryption Suite (AES) for confidential data transport, Rivest-Shamir-Adleman (RSA) for digital signatures and key transport and Diffie-Hellman (DH) for key negotiations and management. Explore special Lightweight protocols like PRESENT and HIMMO. The newer lightweight protocols have an added advantage of being off the beaten track. Alternative public key cryptosystems (APKCs) like Lattice based NTRU provides a level of security comparable to RSA and ECC. However, many IoT devices will have limited access, thus

---

55 Based on notes from Lorie Wigle of McFee/Intel and CISCO Framework for IoT Security
56 http://valerieaurora.org/hash.html
58 The system is now adopted by the IEEE P1363 standards under the specifications for lattice-based public-key cryptography as well as IEEE P1363.1 and ANSI X9.98 Standard for use in the financial services industry.
requiring initial configuration to be protected from tampering, theft and other forms of compromise throughout its usable life, which in many cases could be years.

12.2. Minimal Recommendations

The amount of resources and effort spent on security should be a function of potential damage. Vast majority of IoT Devices are meant to measure and report things and may not require much fortification. McKinsey notes that 40% of the value of IoT is in sharing data. Devices which initiate action and control motors, actuators etc may need more. Potential accident or life threatening potential may require a ground up rethink with hardening against attack as a major Non Functional requirement. A suggested classification with illustration in context of a house/Factory.  

0 - Basic. A casual trespasser can enter if has access.
2 - Secure. Resists simple break in. Padlocks steel doors, Grill windows and unarmed sentry
3 - Hardened. Barriers to intruders, armed sentry, Hot Line to Police.
4 - Ballistic. Fort Knox type against armed intrusion, underground, naval or airborne sneak entry.

For levels 3 /4 it is important to keep pace with technology. Just like ignorance of the law is not a defense, ignorance of hacker arsenal is not a defense.

1. Authentication/Provisioning: Consider LWM2M. In IoT context devices are assumed to be installed in a controlled and secure area within managed premises and do not have much of security built in. M2M assumes machines log into systems and networks using their own credentials and these may be static and hard coded or left to factory defaults. Consider some form of dynamic identity based on a UUID. Explore UUID based controls in devices joining the IoT network if rogue devices are an issue.

2. Authorization: Review your use cases for sensitive operations. See if these should require special authorization over and above a trusted user. Say two users have to confirm or time delay based re confirmation. Build logic for trust level in such request. For example a "card not present" payment request over the web is processed more stringently then if the credit card was swiped at a retail terminal. Use white list of ISP to accept requests. A connection request from

---

59 Consider Level 0 for Sensors in public use like Smart City. Level 1 for consumer and Level 2 or 3 for Industrial against espionage.
60 http://www.slideshare.net/zdshelby/oma-lightweightm2-mtutorial.
61 Hardware id CPU Serial number or MAC may be part of this UUID. Do keep in mind that many providers do not actually pay the IEEE and procure unique MAC ids so some collision is possible.
"unknown" ISP should probably be rejected. Explore ISP capability to patrol and protect connection to IoT facility.

3. **Lightweight Encryption:** Use Cryptographic techniques available in IoT networks. These add 12-30% overhead in terms of compute and (battery) power usage. You may want to use certificate pinning to speed up authentication. In some case self certified certificates may be appropriate Configuring crypto algorithms is complex and many successful attacks have been due to mis-configuration. Review your procedures with experts. Do disable default and backwards compatibility options.\(^\text{62}\)

<table>
<thead>
<tr>
<th>IoT Layer</th>
<th>IoT Protocol</th>
<th>Security Protocol</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application</td>
<td>CoAP</td>
<td>User-defined</td>
</tr>
<tr>
<td>Transport</td>
<td>UDP</td>
<td>DTLS</td>
</tr>
<tr>
<td>Network</td>
<td>IPv6, RPL</td>
<td>IPsec, RPL security</td>
</tr>
<tr>
<td>6LoWPAN</td>
<td>6LoWPAN</td>
<td>None</td>
</tr>
<tr>
<td>Data-link</td>
<td>IEEE 802.15.4</td>
<td>802.15.4 security</td>
</tr>
</tbody>
</table>

**Figure 23:** IoT Stack with standard security options

4. **Super User:** Review use cases for super user or administrators. Provide extra controls and audit logs for usage of these accounts. Consider one time use passwords and session recording for super user access. Integrate with System Information and Event Management (SIAM)\(^\text{63}\) tools if necessary. *Log forensic data for tracing attacks. Review logs periodically and establish alerts. Do not allow repeated attempts to log in.* This is typically a hacker checking out a system. See how 96 bit crypto key was broken because brute force attack exploring 196,607 options was not detected.\(^\text{64}\) *Logging should be at OS level and not easy to bypass.* Adopt a virtual shell or a Honey Pot if needed.\(^\text{65}\)

5. **Controlled access for Operations:** Design programs to use well secured API to call for services. Do not allow access to sensitive hardware or features from different places. The IoT Bridge should be a funnel and choke point for access to IoT devices. For example a WiFi router should not be able to directly address the devices but should have to send a message or call a API to the IoT Hub. Minimize root level shell access. **Isolation between the IoT network and Internet is**


\(^{65}\) [http://conpot.org/](http://conpot.org/)
crucial. Log properly at the IoT hub and IoT Gateway for forensic analysis after an attack.

6. **Static IP**: *Do not keep connected to the internet unless transacting.* Avoid using static IP address as they will get probed by hackers. Waylon Grange talks of having hackers probe a honey pot within hours. Ensure robust firewall and minimal availability of internet protocols. Disable Telnet and SSH. Run minimal and simple programs to interact with internet. Browser can be buggy. Web servers even more so.

7. **OTA**: Consider over the air upgrades carefully. Sign code to avoid malware infection. Schedule updates in segments so as not to overwhelm the network/Gateway and hubs.

8. **Public WiFi are insecure**: Arrange for higher levels of security of WiFi and frequent monitoring of rogue SSID or Access points

9. **Review the top 10 IoT issues** from Open Web Applications Security project.

---
