Applicability of data distribution technologies within ITS

# Scope

A variety of general-purpose data distribution technologies have emerged within the Information and Communications Technologies (ICT) industry. These technologies generally provide services at the Open System Interconnect (OSI) Session, Presentation, and Application Layers (i.e., Layers 5-7). Within Intelligent Transport Systems (ITS), these layers roughly correspond to the Facilities Layer of the ITS station architecture, as defined within ISO 21217.

This Technical Report investigates the applicability of these data distribution technologies within the ITS environment.

# Normative references

# Terms and definitions

facilities layer

ITS station architecture

# Symbols and abbreviations

ICT

ITS

OSI

# Transitioning from traditional to cooperative thinking

## General

ITS is heavily dependent upon the exchange of varied types of data between and among disparate types of physical objects. Physical objects include:

* Centres (e.g., fixed-location facilities and cloud-based back-office services)
* Field devices (e.g., devices along the roadside)
* Vehicles
* Travelers (e.g., personal devices)
* Support systems (typically fixed or back-office, that provide services enabling ITS, but do not directly provide ITS services)

The data that these systems exchange include:

* Live elemental data (e.g., vehicle speed, location, signal timing information, etc.)
* Live aggregated data (e.g., average speeds, rain rates, etc.)
* Status information (e.g., status of reversible flow lanes)
* (Relatively) static data (e.g., map information)
* Exceptional reports (e.g., information on traffic incidents, realignment of lanes due to incidents or road work, etc.)
* Configuration data (e.g., certificate revocation lists, traffic regulation information, software configuration, etc.)
* Coordination data (e.g., exchanges to coordinate a response plan among centres)
* Traffic regulations
* Software updates (e.g., for on-board applications)
* Security certificate and revocation list distribution

The varied data exchanges among the different physical objects also has various needs for data distribution. For example, software updates might be intended for specific vehicles. Traffic regulation data is likely intended for all vehicles within a jurisdiction. Exceptional reports might be intended for vehicles approaching an incident. And finally, there is an increasing appreciation that some of the information exchanged might be useful to support ITS services other than the ITS service for which the data was originally intended.

There are a variety of technical and institutional challenges in successfully sharing data in a timely and secure manner. Challenges include:

1. Acquiring the data (e.g., through sensors)
2. Defining ownership and access rights for the data
3. Securing the data (e.g., authentication, authorization, confidentiality, integrity, availability, etc.)
4. Achieving adequate market penetration of lower-layer communication technologies
5. Agreeing on the upper-layer protocols for exchanging the data over the communication technologies
6. Standardizing the definition of data for use in various contexts
7. Defining performance criteria for different uses of the data
8. Maintaining the interface over the life cycle of the involved physical objects. Operational lifetimes for ITS devices vary radically; field devices often have lifetimes of 15-20 years, vehicles closer to 10 (though often much longer), and smartphones merely 18-24 months.

This technical report focuses on the upper-layer protocols (i.e., item 5) while recognizing that this layer will need to provide adequate services to support the other issues. For example, part of our analysis of the data distribution technologies considers the ability of each technology to provide authentication services that meet rigorous ITS demands as well as an analysis of the performance implications of each technology (e.g., processing and bandwidth requirements). Other issues listed are largely left to other stakeholders in the ITS community.

## Systems engineering process

The systems engineering approach to designing any complex system is to work with the relevant stakeholders, including service providers and system integrators, to develop a “Concept of Operations”, or ConOps. This involves describing in detail the service (the “why”), the actors participating in the service (the “who”), and the requirements on information that must be generated and exchanged by entities engaged in the service (the “what”).

Once agreement is reached on the ConOps, the implementers work together to develop a high-level design (i.e., an architecture) that defines the means by which the service will be implemented (the “how”), which must (directly or indirectly) define the details of how the information is encoded and transferred between physical objects. If the system is intended to support an open interface (i.e., so that competing manufacturers can interoperate); these design details should be defined within open standards and developed with broad-based consensus.

## Traditional silos versus cooperative approach

Once the architecture is developed, each interface is designed by its own group of experts to meet the defined needs. However, this division of effort tends to produce “silos” of thought that can often result in four major problems:

1. **Competing protocol selection:** Different silo efforts are likely to select different approaches to exchanging data. There are many off-the-shelf protocols that can be extended to support most ITS data exchange needs and some experts may wish to develop their own protocols to optimize performance in certain cases. While each decision may be reasonable in isolation, each protocol adopted by the ITS industry has costs associated with stakeholders learning the technology, implementers programming with the technology, testers verifying conformance to the technology, and maintenance issues with maintaining backwards compatibility, as well as memory and processing issues within devices that have to support multiple technologies. Ideally, the ITS community as a whole should attempt to identify a suite of preferred protocols that meet industry needs so that the variability in systems is minimized.
2. **Competing data definitions:** Different silo efforts are likely to produce different data definitions to describe the same real-world conditions. This greatly complicates data sharing, increases potential translation errors, and increases integration costs. Ideally, all ITS data definitions should be developed in a cooperative fashion.
3. **Limited scope and lack of forwards compatibility:** Engineers within the silo teams will often attempt to “optimize” their design; however, without a complete knowledge of how data might be used, it is impossible to know if a design is truly optimal or not. This can partially be overcome by ensuring that the reference architecture is developed with a broad as scope as practical, but since innovations occur over time, it must be understood that no effort will ever be omniscient about how the data might be used; we can only attempt to consider as much data as possible.
4. **Competing efforts:** A final challenge facing any development team is that there is often different competing and/or overlapping efforts across the world. Once standards are developed, it is often difficult and expensive to harmonize the results after the fact.

This technical report attempts to address the first issue by identifying different protocols that have been suggested for use within the ITS industry, comparing their respective characteristics, and suggesting a preferred set of protocols for future use.

## Summary of needs and considerations

Within the scope of this document, data distribution is defined to be a service that maintains a directory of users and facilitates the delivery of select data to those users. In order to evaluate specific technologies, the analysis began by identifying the key stakeholder needs and considerations for data distribution. While the needs of ITS vary based on the specific information flow considered, an analysis of the Harmonized Architecture Reference for Technical Standards (HARTS, available at http://htg7.org) indicate that the industry needs data distribution technologies that can support:

* Confidentiality: Many of the defined communications require encryption meeting the requirements of FIPS 140-2 level 3. In some cases, the confidentiality must be maintained to the application entity (i.e., meaning the data distribution technology should not be able to decrypt the information). Additionally, some cases require intrusion detection and mitigation functions that can inspect data distribution messages.
* Integrity: Virtually all flows should have integrity meeting the requirements of FIPS 140-2 level 3 that ensures that received information is authenticated and authorized. In some cases, intrusion detection and mitigation functions that can inspect data distribution messages are also required.
* Availability: Some information flows require support for multiple communication technologies to allow communications when the primary communication channels are unavailable.
* Latency: While most information flows within the architecture allow for up to 2 seconds of delay between production of the data and its consumption; there are a relatively small number of flows where ultra-low latency (100 ms) is required. Support for the ultra-low requirement could be handled by other means if needed.
* Throughput: For most flows, the data distribution technology should be able to deliver at least 10 kb/s of aggregate data subscriptions from a single ITS-SU source. In a few cases (5-10% of information exchanges), the data distribution technology needs to be able to deliver up to 500 kb/s of aggregate data subscriptions from a single ITS-SU source.
* Pseudonymity: For most data distribution flows, pseudonymity is not required (i.e., it is acceptable or even desired for the receiver to be able to identify the source).
* Quality of Service: The data distribution technology should provide at least a high level of assurance that the data throughput expectations will be met under all conditions, and in some cases, it must be guaranteed.
* Communication technology: The data distribution technology should be readily deployable using range of IP-based communication technologies.
* Non-repudiation: The data distribution technology should be able to support non-repudiation services such that the sender of a message is not able to successfully claim that it did not send it.
* Misbehaviour reporting: The data distribution technology should be able to report any misbehaving actors to the appropriate systems to ensure that all systems can be properly prepared.
* Geofencing: In several environments it is useful to restrict publications to a specific geofenced area and/or to restrict publication content to information related to a geofenced area.
* Flow filters: In many cases, it is desirable to allow a subscriber to request topic publications to be filtered to better meet its needs based on frequency, accuracy and other parameters. For example, while a source may provide once-per-second updates, a subscriber may only need and want once-per-minute updates.
* Efficient repackaging: In some cases, it may be advantageous if the data distribution technology is able to combine topics from multiple updates from one or more sources and package them into a single publication to meet the needs of each user.

## Solution characteristics

Each solution is also characterized by a number of other factors as follows.

### Architectural topology

Part of the goal in sharing data among systems is to minimize the complexity associated with maintaining connections between the various components. Each data distribution technology is based on a architectural topology that can generally be grouped into one of four styles as described below.

#### Mesh Topology

Within a mesh topology, every application entity is required to establish a connection with every other application entity with which it wants to communicate. Once a connection is established, the two applications can subscribe for information and provide publications as necessary.

The mesh topology is depicted in Figure 1.

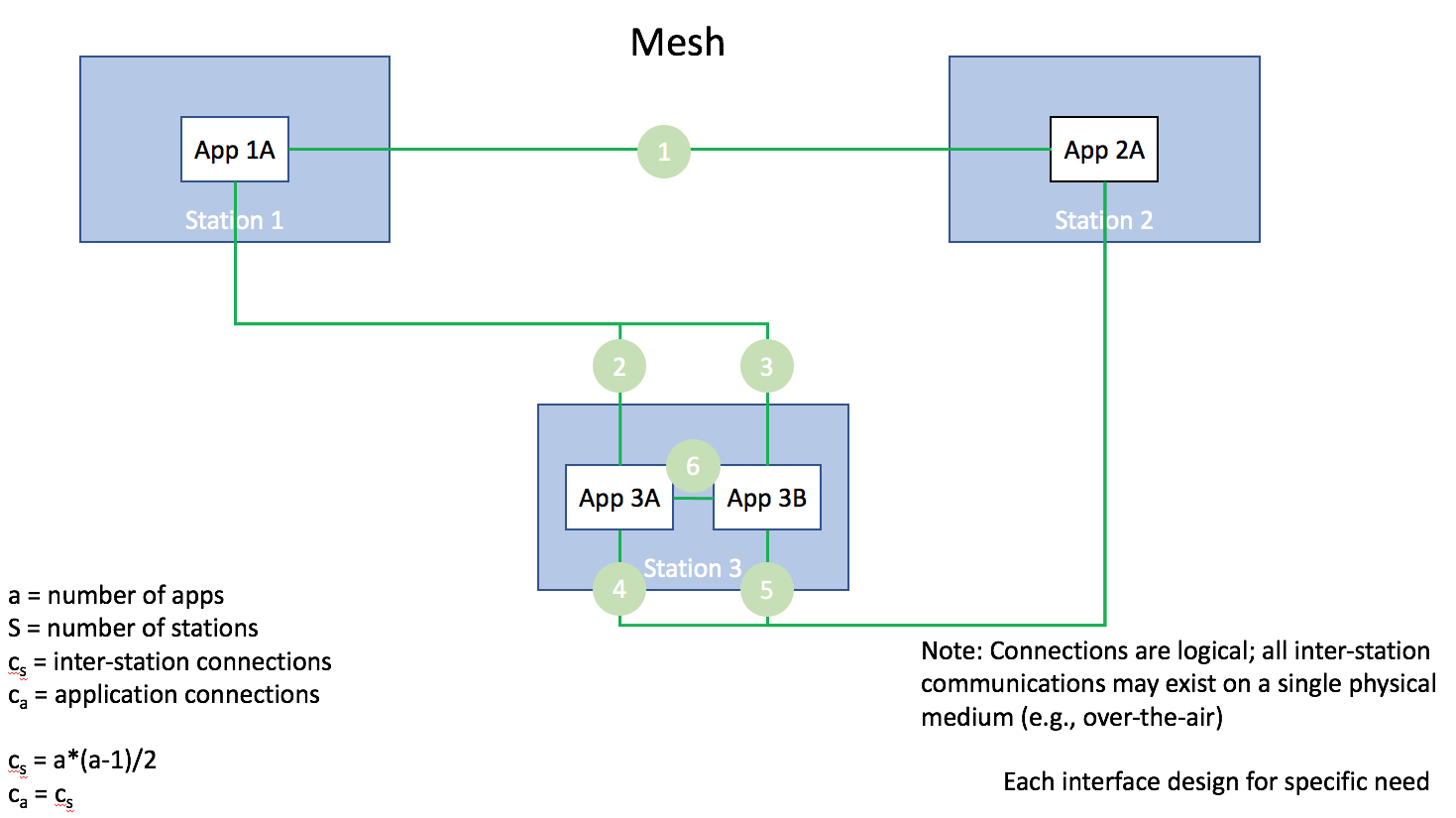


Figure 1: Mesh topology

The mesh topology has the advantage that an application providing data can ensure that the application requesting the data is authorized to receive it; but this also means that each application has to spend resources managing connections and authorizing requests. This can be especially challenging in a cooperative environment where requesters are not necessarily part of a pre-defined list and the number of connections are not necessarily constrained.

#### Hub-and-spoke topology

Within a hub-and-spoke topology, every spoke application entity is required to establish a connection with a hub application. The spoke can then subscribe for information or publish information to the hub. The hub then has the responsibility of forwarding the publications to all applications that have subscribed for the data.

The hub-and-spoke topology is depicted in Figure 2.

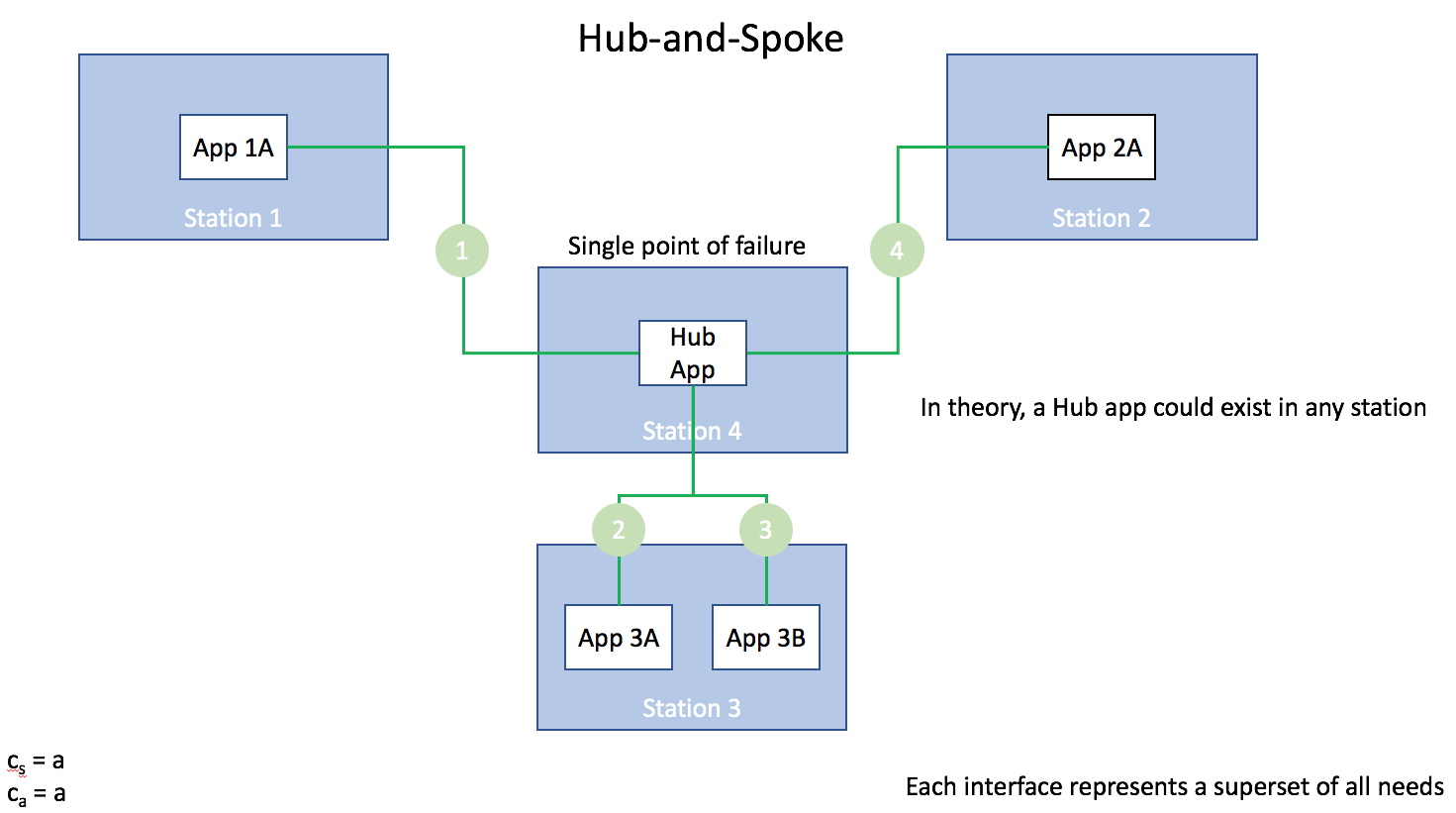


Figure 2: Hub-and-spoke topology

The hub-and-spoke topology has the advantage that an application providing data can focus on providing its core service while managing a single connection; however, it delegates the authorization task to a remote hub application, which potentially raises issues in a C-ITS environment where the hub application is a separate system (i.e., owned and/or operated by a different legal entity and therefore increasing the number of legal entities with theoretical access to the data). The design also presents challenges for a constantly changing network where devices are mobile and are constantly connecting and disconnecting.

#### Databus service topology

Within a databus service topology, every device supports its own service that acts in a manner similar to a hub. Each application within each device connects to its local hub service. The hub service then connects to the hub services in other devices. Applications publish information to their local hub; the hub service then takes care of distributing the information to other local entities and remote hub services that are authorized.

The databus service topology is depicted in Figure 3.

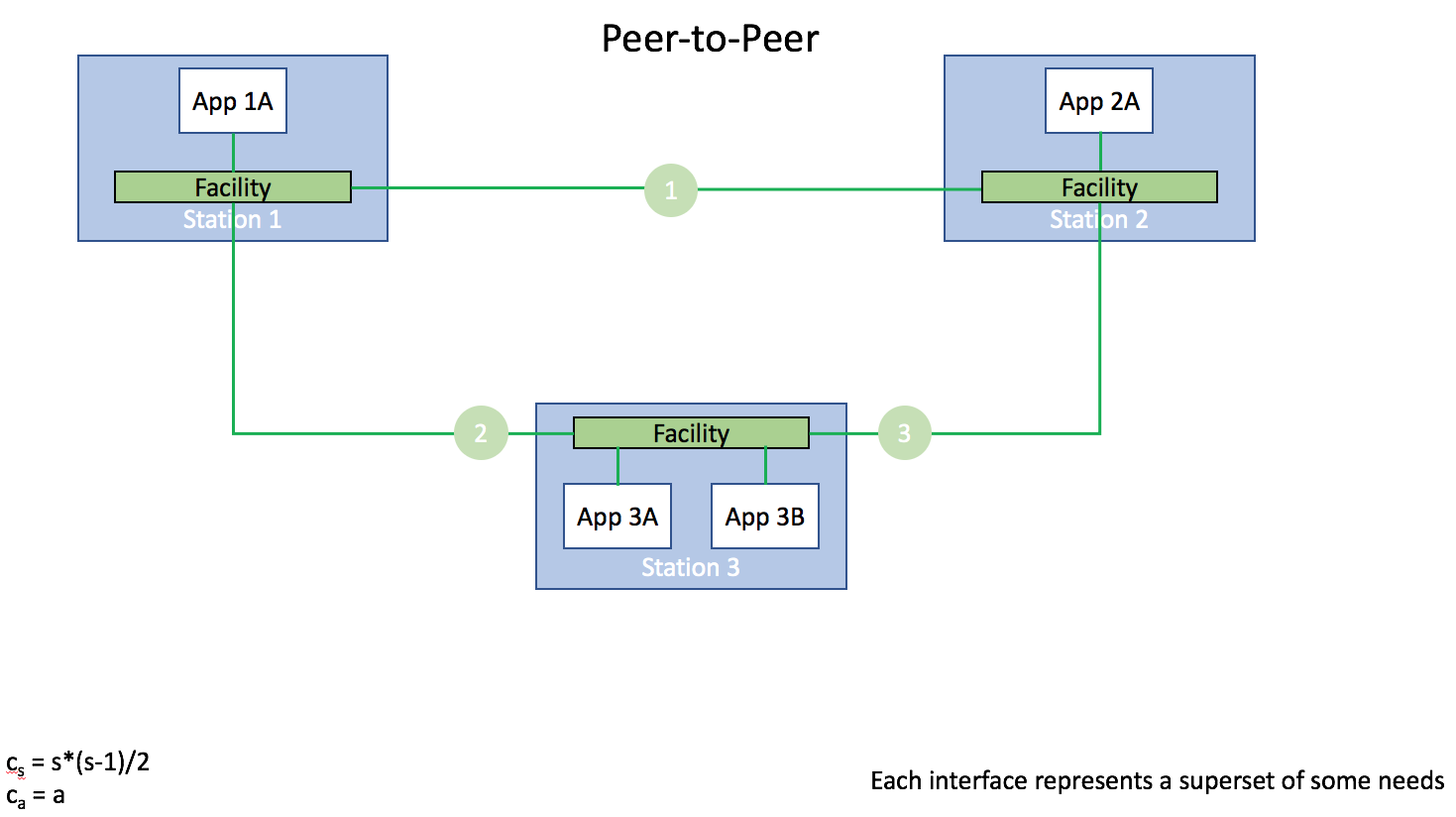


Figure 3: Databus service topology

The databus service topology has the advantage that an application providing data can focus on providing its core service while managing a single connection; further, the authorization task is still largely controlled by a local service within the same system. While a portion of the authorization task is the responsibility of the remote hub service, data will only be sent to the remote service if the remote service (and hence that system) has authorization. The biggest challenge for this design in in maintaining connections in the mobile devices, but this is less of a problem than in some other designs since there are fewer connections to maintain and the management of these connections are concentrated in dedicated software.

#### Hierarchical hub topology

The hierarchical hub topology combines the concepts of the hub-and-spoke and databus service topology. Every device supports its own service that acts in a manner similar to a hub; and the various devices also connect via a device hub. Each application within each device connects to its local hub service. The hub service then connects to a central device hub. Applications publish information to their local hub; the local hub service then takes care of distributes information to the device hub, if authorized. The device hub then distributes to end applications via their own local hubs.

The hierarchical hub topology is depicted in Figure 4.

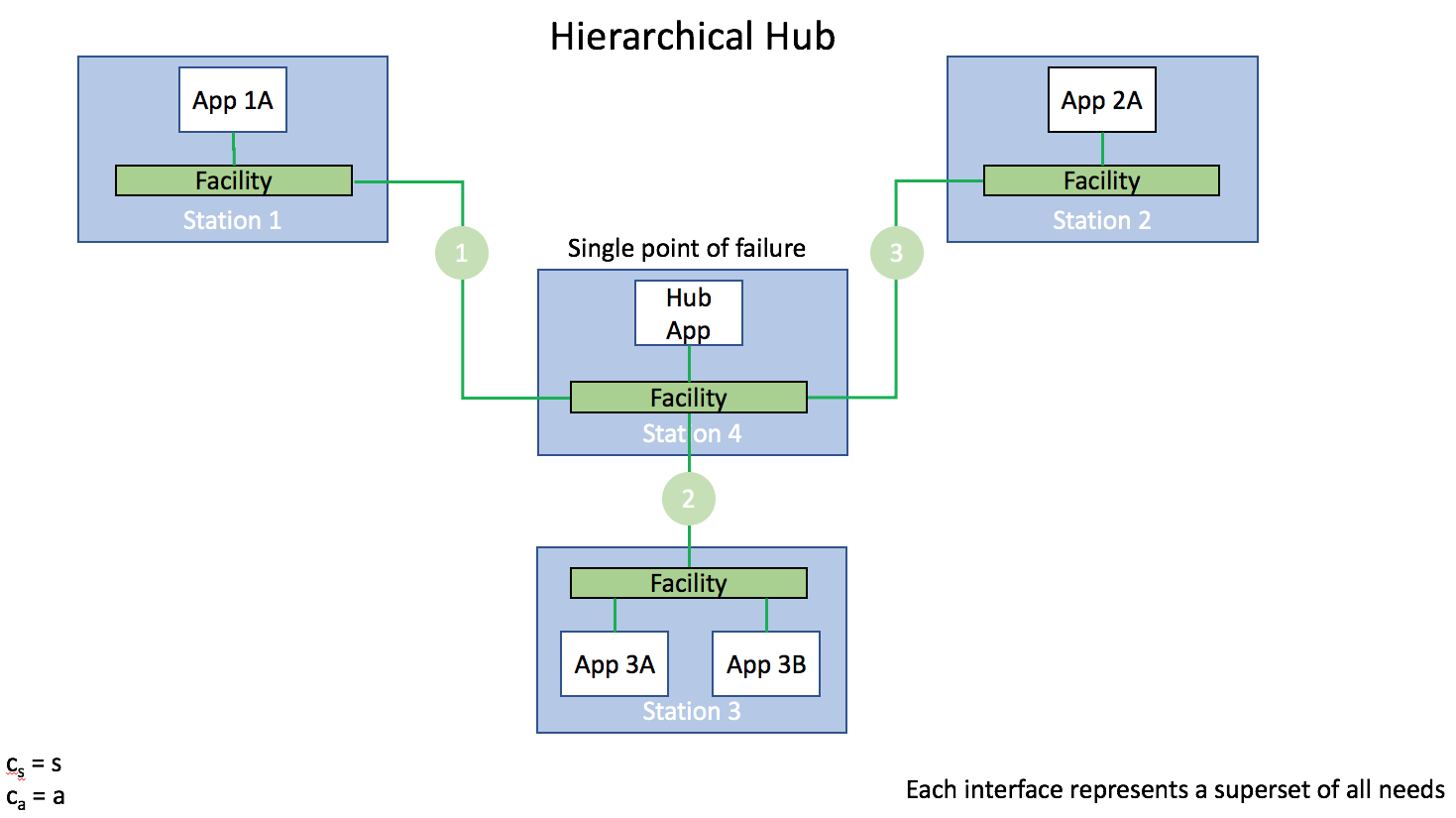


Figure 4: Hierarchical hub topology

The hierarchical hub topology has many of the advantages of both the hub-and-spoke and peer-to-peer topologies.

#### Summary of topologies

The various advantages and disadvantages of each topology are summarized in Table 1.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Characteristic | Mesh | Hub-and-spoke | Peer-to-peer | Hierarchical hub |
| Connections | Application-to-application | Application-to-hub | Facility-to-facility | Facility-to-hub |
| Code Size | High | High | Low | Low |
| Processing | High | Low | Moderate | Low |
| Single point of failure | No | Yes | No | Yes |

### Maturity

Each technology is characterized by the following maturity characteristics:

* Standardization Status: Indicate the standard organization(s), standard number, date approved (or current draft status)
* Time in market place: Time since the first version of the specification was deployed
* Suppliers: Identify specific products that can be used to develop an ITS implementation that conforms to the standard (we might change this to a simple count in the published report)
* ICT deployments: List example uses of the technology outside of ITS
* ITS deployments: List specific uses of the technology within ITS
* Notes: Any other relevant information

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Technology | Topology | Status | Introduced | Suppliers | ICT | ITS | Notes |
| AMQP | Database Service | OASIS AMQP | 2011 |  |  | EU? Scandavia? |  |
| Apache Kafka | Database Service | Open source | 2012 |  |  | Wyoming deployment |  |
| ISO 17429 | ? | ISO TS | 2017 |  |  |  | ISO/TC 204 standard |
| MQTT | Database Service | OASIS MQTT | 2013 |  |  | Queensland, Australia (CAVI) |  |
| OMG DDS | Databus Service | OMG DDS | 2004 |  |  |  | RTI |
| SNMPv1 | Mesh | RFC 1157 | 1990 | <10 | Manage network devices | Manage ITS field devices (e.g., message signs) | Publish-subscribe is not directly supported |
| SOAP (with DATEX II or TMDD) | Mesh | W3C SOAP | 2000 (with ITS pub/sub ~2008) | Generally custom | Structured information for web services | Traffic management data |  |

### Dependencies

* UDP: Indicate if the protocol requires a connection-oriented stack (e.g., TCP) or can work connectionless (e.g., UDP)
* OS: List the target OS supported by the current products
* Languages: List the programming languages that current products interface with
* Firewalls: Indicate the level of effort required to set up communications through firewalls

### Impacts

* Device deployment: Indicate what requirements this assumes on system devices. For example, do all devices have to support this technology; can gateways be used; etc.
* Infrastructure topology: What implications does this have on network topology. For example, does a hub have to be present and if so, can this be overcome in areas where there is not an infrastructure hub available
* Data definitions: Language(s) used to describe the data (e.g., ASN.1, XML, IDL, UML) and are there automated ways to convert among them
* Scalability: Can the protocol be stripped down to its basics so that it can be implemented on a simple device easily

### Performance-based analysis

* Reference: Provide information on the processing and communications load for live operations

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Technology | UDP | OS | Languages | Firewalls | Device Deployment | Topology | Data Definitions | Performance |
| AMQP |  |  |  |  |  |  |  |  |
| Apache Kafka |  |  |  |  |  |  |  |  |
| ISO 17429 |  |  |  |  |  |  |  |  |
| MQTT |  |  |  |  |  |  |  |  |
| OMG DDS |  |  |  |  |  |  |  |  |

### Capabilities

* Confidentiality: Many of the defined communications require encryption meeting the requirements of FIPS 140-2 level 3. In some cases, the confidentiality must be maintained to the application entity (i.e., meaning the data distribution technology should not be able to decrypt the information). Additionally, some cases require intrusion detection and mitigation functions that can inspect data distribution messages.
* Integrity: Virtually all flows should have integrity meeting the requirements of FIPS 140-2 level 3 that ensures that received information is authenticated and authorized. In some cases, intrusion detection and mitigation functions that can inspect data distribution messages are also required.
* Availability: Some information flows require support for multiple communication technologies to allow communications when the primary communication channels are unavailable.
* Latency: While most information flows within the architecture allow for up to 2 seconds of delay between production of the data and its consumption; there are a relatively small number of flows where ultra-low latency (100 ms) is required. Support for the ultra-low requirement could be handled by other means if needed.
* Throughput: For most flows, the data distribution technology should be able to deliver at least 10 kb/s of aggregate data subscriptions from a single ITS-SU source. In a few cases (5-10% of information exchanges), the data distribution technology needs to be able to deliver up to 500 kb/s of aggregate data subscriptions from a single ITS-SU source.
* Quality of Service: The data distribution technology should provide at least a high level of assurance that the data throughput expectations will be met under all conditions (e.g., even with data packet losses), and in some cases, it must be guaranteed.
* Communication technology: The data distribution technology should be readily deployable using range of IP-based communication technologies.
* Non-repudiation: The data distribution technology should be able to support non-repudiation services such that the sender of a message is not able to successfully claim that it did not send it.
* Misbehaviour reporting: The data distribution technology should be able to report any misbehaving actors to the appropriate systems to ensure that all systems can be properly prepared.
* Geofencing: In several environments it is useful to restrict publications to a specific geofenced area and/or to restrict publication content to information related to a geofenced area.
* Flow filters: In many cases, it is desirable to allow a subscriber to request topic publications to be filtered to better meet its needs based on frequency, accuracy and other parameters. For example, while a source may provide once-per-second updates, a subscriber may only need and want once-per-minute updates.
* Efficient repackaging: In some cases, it may be advantageous if the data distribution technology is able to combine topics from multiple updates from one or more sources and package them into a single publication to meet the needs of each user.
* Registration and discovery: Indicate if the technology automatically discovers available systems and information and registers itself with others
* Multicast: Indicate if the technology supports notifying multiple subscribers of data through a single publication and whether this is achieved via true multicast or broadcast
* Support for ITS security mechanisms: Indicate if the technology natively supports cooperative ITS security (e.g., IEEE 1609.2); if not can existing products be readily configured to support this feature in an interoperable fashion; if not what would be required to support this level of security
* Configurability: Ability to manually adjust parameters to adjust memory usage, processing usage; what are the advantages, how hard is it, etc.
* System management: What information and tools are available to manage performance of the network

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Technology | Confidentiality | Integrity | Availability | Latency | Throuughput | QoS | Non-repudiation |
| AMQP |  |  |  |  |  |  |  |
| Apache Kafka |  |  |  |  |  |  |  |
| ISO 17429 |  |  |  |  |  |  |  |
| MQTT |  |  |  |  |  |  |  |
| OMG DDS |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Technology | Misbehaviour | Geofencing | Flow Filters | Efficiency | Discovery | Multicast | ITS security |
| AMQP |  |  |  |  |  |  |  |
| Apache Kafka |  |  |  |  |  |  |  |
| ISO 17429 |  |  |  |  |  |  |  |
| MQTT |  |  |  |  |  |  |  |
| OMG DDS |  |  |  |  |  |  |  |

Gateway availability

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | AMQP | Apache Kafka | ISO 17429 |  |  |  |
| AMQP |  |  |  |  |  |  |
| Apache Kafka |  |  |  |  |  |  |
| ISO 17429 |  |  |  |  |  |  |
| MQTT |  |  |  |  |  |  |
| OMG DDS |  |  |  |  |  |  |