Integrating RFID Readers in the Enterprise IT – Overview of Intra-organizational RFID System Services and Architectures

Christian Floerkemeier
Auto-ID Labs
Massachusetts Institute of Technology
Cambridge, MA 02139
floerkem@mit.edu

***Abstract*— RFID system deployments require the configuration, monitoring and data management of RFID readers. This paper provides an overview of different services deployed in intra-organizational RFID systems and analyzes system architectures implemented today. We also compare emerging standards developed by the EPCglobal community that aim to standardize system interfaces and reader protocols in RFID deployments. Our analysis distinguishes a centralized architecture, where a controller device or a software component on an application server locally controls the RFID readers, and an autonomous architecture, where the RFID readers execute application logic and are managed by enterprise IT management systems.**

I. Introduction

As the use of RFID systems increases, large scale deployment of RFID are highlighting certain challenges. The management and control of large number of RFID readers is becoming an issue from a network administration perspective. This includes monitoring the health of RFID readers and maintaining a consistent configuration across all RFID readers. The shared nature of the wireless medium may require coordination among the RFID readers to minimize interference and comply with local radio regulations. Readers also do not operate independently but are sometimes triggered by external sensors that need to be configured.

Large number of RFID readers also create significant data volumes. The data captured by the RFID readers need to be filtered and aggregated due to the presence of redundant and unwanted data. Most important, the captured RFID data have to be interpreted in an application context to make the data capture meaningful in the first place.

There are a number of different (partly proprietary) solutions to address some or all of the above mentioned challenges. This paper presents a taxonomy of the corresponding system services and architectures. We aim to provide a comprehensive understanding of common industry practices and thus facilitate future deployments and possible standardization activities. We also discuss how these system architectures relate to emerging standards and how these standards could possibly be enhanced to support future RFID deployments.

The paper is organized as follows. Section II discusses related work. Section III presents a number of different services and components that are commonly found in an RFID deployment. In Section IV, we discuss the services offered by today's RFID reader products. Section V presents different system architectures that we identified. In Section VI, we discuss EPCglobal specifications that aim to standardize system interfaces. Before we conclude in Section IX, Section VII discusses the adoption of today's EPCglobal reader protocol standards and Section VIII presents potential enhancements to existing specifications.

II. Related Work

There are a number of previous publications that discuss requirements towards an RFID infrastructure and propose system architectures [1]–[4]. The work presented in this paper differs from previous work because we do not propose a particular system architecture but compare different system architectures. We also focus in particular on emerging standards in our analysis.

Within the EPCglobal community, technology vendors and end users that deploy the RFID technology have been working jointly on the development of specifications that standardize the interfaces between RFID tags, readers and enterprise IT systems. The *Architecture Framework* published by EPCglobal [5] provides a comprehensive overview of the EPCglobal standards. It shows how the different interface standards are related and outlines the principles that have guided the design of the standards. The *EPCglobal Architecture Framework* does not dictate a particular system architecture, but leaves this to implementers who can choose the system architectures that are most appropriate for their deployments. The analysis presented in this paper is thus complementary to the *EPCglobal Architecture Framework* because it provides insights into which system architectures are actually deployed. The *Architecture Framework* document distinguishes a number of different roles, such as *RFID Reader*, *Filtering & Collection* and *Capture Application*, each of which implements a set of different services. Rather than classifying the services as roles which are implemented on a particular device as in the *EPCglobal Architecture Framework*, we categorize them as base, configuration, monitoring and data processing services.

In WLAN access point deployments, the management, monitoring, and control of large numbers wireless access points also represents a challenge. In the WLAN domain, there are a number of different approaches to administer WLAN access points as discussed in [6]. This paper shows

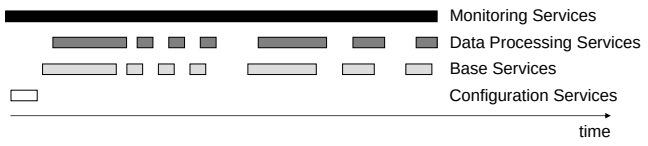


Fig. 1. Timeline of RFID operation featuring base, configuration, monitoring and data processing services.

that today's RFID system architectures have some similarities with the WLAN access point architectures. However, the paper also shows that RFID systems offer a number of services that do not have equivalents in the WLAN domain, such as application dependent data processing. In WLAN, the need for central control stems from wireless node mobility and network security, neither of which apply in the same way to RFID. In this document, we also leveraged some of the concepts introduced in the WLAN domain, such as the distinction of different service sets.

III. RFID System Services

In this section, we discuss services provided by different components within an RFID system. This includes data and device management, control, and 'over-the-air' services. In the remainder of the paper, we will show how these services are provided by different entities in different RFID system architectures. Our analysis of the RFID system services begins with the base service set (BSS) specified in the air interface communication standards. These air interface standards that specify the 'over-the-air' interface between one or more transponders and a single reader [7], [8] define the services listed in Table I. In order to facilitate these services, the air interface standards also specify physical layer properties such as coding, modulation, timing, and data link layer properties such as medium access schemes. These base services (BSS) are in practice triggered in a number of different ways, e.g. by external sensors, by external applications, by timers on the readers, by the detection of certain tags, and by humans who press a button on a handheld.

This BSS is supported by a number of additional services that configure, control and monitor the system components that carry out these services and process the data captured. We distinguish different service sets according to different phases of the RFID operations (cf. Fig. 1): Configuration services are executed before any base services are executed, monitoring services are running while base services are executed and data processing services are executed once the base services returned captured data. In case of memory access and deactivation, the processing of the captured tag ids can trigger subsequent base services such as tag memory access.

Prior to any data capture, the networking and radio module embedded in a reader needs to be configured appropriately. The configuration service set (CSS) is responsible for setting network parameters, such as IP addresses, but also RF parameters such as transmit power and frequency channel and air interface protocol-specific parameters, such as timing

TABLE I

Base 'Over-the-air' Service Set (BSS).

Service	Description
Transponder Singulation	Collects the identification numbers (ID) of (selected) transponders in range
Transponder ID Programming	Writes identification numbers to transponders
Transponder Memory Access	Reads from and writes to the general purpose memory on a transponder
Transponder Deactivation	Disables the transponder for privacy reasons

TABLE II

Configuration Service Set (CSS).

Service	Description
Network Interface Configuration	Discovers and sets reader networking parameters and identity, e.g. the IP address
Firmware Management	Distribute and manage firmware version on readers
Antenna, Tag Population & Memory Selection	Specify reader antennas and tag population to be inventoried. In case of tag memory access, specifies memory fields to be accessed
Base Service Set Scheduling	Sets how different BSS services, such as tag inventory, access, and deactivation, are triggered and stopped
RF Transmitter Configuration	Sets transmit channel, hop sequence, transmit power for readers
Air Interface Protocol-Specific Configuration	Configures timing, coding and modulation parameter of a specific air interface protocol on the readers

and coding parameters (cf. Table II). The configuration phase might also comprise specifying which base reader services such tag identification or tag memory access are executed upon the appropriate triggers. Readers can also be configured to only use certain antennas and select a particular tag population over the air interface. The latter avoids that the unwanted data need to be filtered out post-capture (cf. Fig. 2).

Parameters characterizing the network interface or regulatory region are configured once and unlikely to change during the operation of the RFID reader. However, there are also some configuration parameter that will be changed frequently depending on the type of application. Examples include transmit power adaptation for distance estimates, transmit channel changes for interference avoidance and tag population selection to avoid reading a 'parked RFID tag' continuously. The result is that the frequency at which the reader configuration needs to be modified and the resulting coupling between base and configuration services is use case

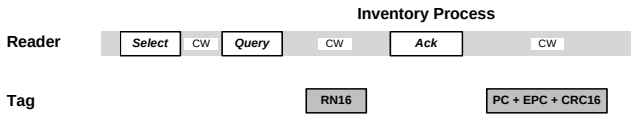


Fig. 2. "Select" command in the EPCglobal UHF Class 1 Generation 2 Protocol (ISO 18000-6C) [7]. A particular tag population is selected before the inventory process is initiated with the "query" command.

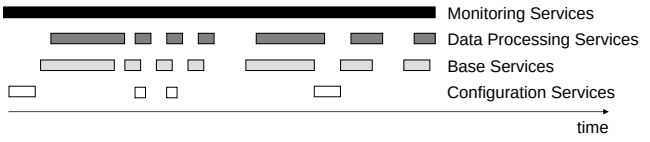


Fig. 3. Timeline of RFID operation featuring base, configuration, monitoring and data processing services with frequent changes to the reader configuration.

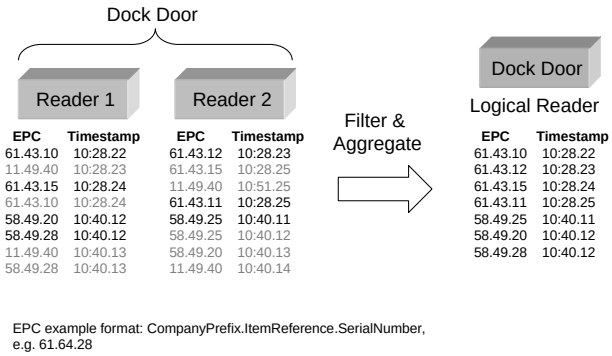


Fig. 4. Filtering and aggregation of RFID data: data of the two dock door readers are combined, duplicate EPCs are eliminated, EPC 11.49.40 is filtered out, and quantities of product categories are calculated. Eliminated 'reads' are shown in grey.

dependent. On the one hand, there are those applications that are unlikely to require changes to the initial configuration of the reader (cf. Fig. 1). After the initial configuration, the reader executes base services and asynchronously notifies the data consumers about tag data captured. Such applications allow for the separation of the control and data plane. On the other hand, there are those applications that require tight coupling between base service execution and configuration services with frequent updates to the configuration (cf. Fig. 3). In the latter case, the separation of control and data plane is not feasible.

The data processing services (DPSS) include services that clean the data captured by filtering out tag IDs of no interest to applications and computing aggregates over the tag data captured. This includes aggregates in the time domain, where entry and exit of tags in the read range are determined, and aggregates in the space domain, where the data captured across multiple reader antennas or even readers are computed (cf. Fig. 4). Since there is frequently some uncertainty about the true location and movement of an RFID transponder relative to the reader, there are additional services that eliminate so-called 'false positive' reads and that possibly even estimate the movement of RFID transponders. Other services include tag data translation and persistent storage.

Messages exchanged between different distributed services travel over a number of communication nodes. This means that some messages may be lost in transit. Additionally, it is possible that either the recipient's or the sender's system fails while a message is in transit, leaving the system in a state of confusion as to whether a given message has been processed or not. Reliable messaging protocols provide guaranteed end-to-end delivery of messages. Reliable messaging refers to the ability to deliver a message once and only once to its intended receiver, to deliver messages in order, and to make the failure to deliver a message known to sender and receiver.

The DPSS also includes services that interpret the RFID data captured in an application context to generate the corresponding application events. For a supply chain application, this might include matching the detected tag identifiers against a list of identifiers in an electronic advance shipping notice. The result of the data interpretation can be the generation of a business event such as 'Shipment complete' to an enterprise resource planning system or a immediate feedback to local staff via a display. The application logic execution service is typically utilized after the other DPSS services such as filtering, aggregation and tag identifier translation preprocessed the captured RFID data.

While filtering, aggregation and tag data translation typically uses predefined operators, data interpretation often relies on custom developed application logic that processes the incoming RFID data. This results from the significantly broader scope of this service when compared to aggregation or filtering. However, there have been a number of (commercial) efforts to define standard workflows for typical RFID application such as dock door receiving. While standardized workflows can reduce the amount of custom application software development required for each deployment, variations from the standardized workflow in real-world processes typically still result in customization and additional software development.

It is worthwhile to mention that the 'filtering' can also be performed by limiting data capture to a subset of reader and reader antennas in the first place and by selecting a specific tag population (cf. Table II and Fig. 2). The filtering is then effectively carried out over the air interface by configuring the reader appropriately. In ISO 18000-6C, this is achieved by executing one or more 'Select' commands before an inventory round is initiated with a 'Query' command. Since tags might miss a 'Select' command that deselects them and only receive the subsequent 'Query' command, 'filtering' is performed over the air interface as well as in software. Even aggregation in the time domain can be performed over the air interface using advanced features in air interface protocols such as persistent inventory flags in ISO 18000-6C. In many applications, RFID tags are still identified multiple times while they are in the read range due to multipath effect and it becomes a necessity to compute entry/exit aggregates in software.

There are also a number of different monitoring services (MSS) that observe the health of the reader, the RF environment and the network connection to the reader (cf. Table IV). These monitoring services are essential to operate large reader deployments reliably. Services include heartbeat messages exchanged between reader and monitor to detect

TABLE III
DATA PROCESSING SERVICE SET (DPSS).

Service	Description
Filtering	Removes unwanted tag identifiers from the set of tag identifiers captured, e.g. based on the product type or manufacturer encoded in the identifier.
Aggregation	Computes aggregates in the time domain (entry/exit events) and the space domain (across reader antennas and readers) and generates the corresponding 'super'-events.
Identifier Translation	Translates between different representation of the identifier, e.g. from raw tag object identifier in hexadecimal format to EPC in URN notation
Persistent Storage	Stores RFID data captured for future application requests
Reliable Messaging	Allow RFID data to be delivered reliably in the presence of software component, system and network failures
Location/Movement Estimation	Detects false positive reads of far-away tags that are outside the 'typical' read range and estimate the direction of movement
Application Logic Execution	Interprets the RFID data captured in an application context and generate the corresponding application events, e.g. detect whether a shipment is complete

TABLE IV
MONITORING SERVICE SET (MSS).

Service	Description
Network Connection Monitoring	Check that the reader can communicate captured RFID data over the network
RF Environment Monitoring	Check RF noise and interference levels to safeguard reliable identification operation
Reader Monitoring	Check that the reader is up an running and executing BSS as configured for example via monitoring the number of successful/-failed read and write operations

network failures, the monitoring of antenna status, memory overflows and reboot alarms, and RFID interference updates from the reader to the monitor.

IV. READER CAPABILITIES

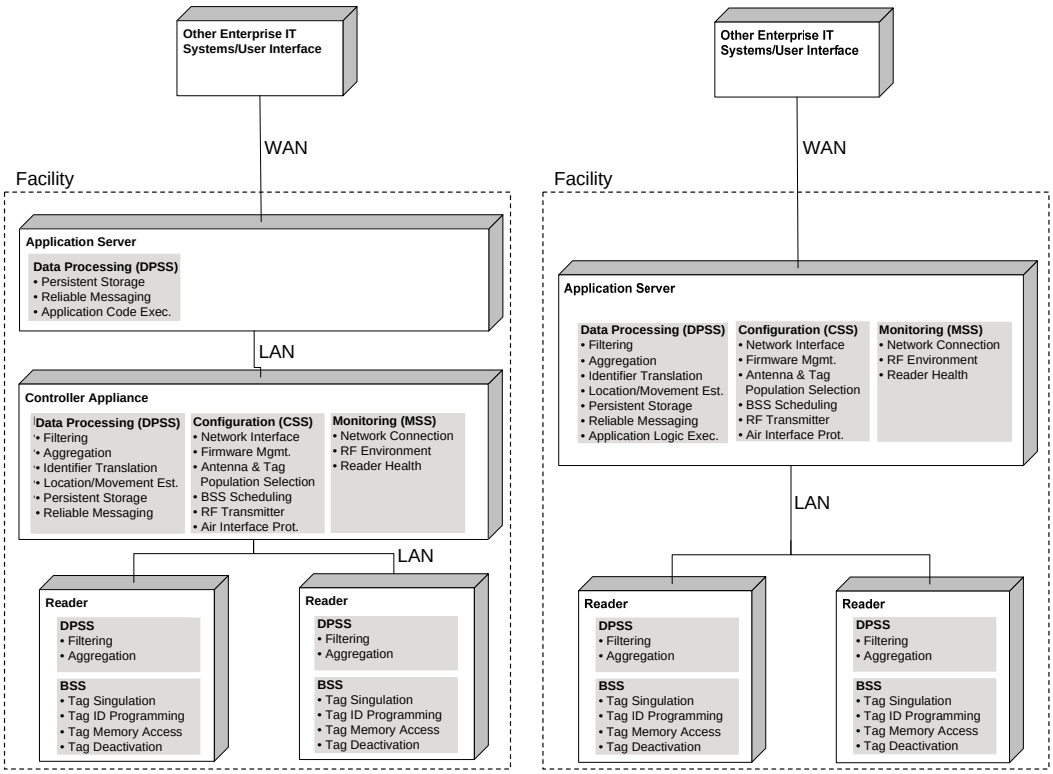
Before we present different RFID system architectures that provide the services listed in the previous sections, we discuss the different RFID reader categories available on the market today. This is important because the different architectures are heavily influenced by the type of readers deployed.

A typical reader is comprised of a radio module, a general purpose computing module, a network interface, and general input/output pins. The general purpose computing unit can be a low-end microcontroller or an embedded processor with significant computing resources. All readers provide the base service set mentioned earlier. Depending on the capabilities of the general purpose computing unit, different reader types provide different data processing services. While one frequently distinguishes 'dumb' and 'intelligent/smart' readers or 'thin' and 'fat' readers to categorize readers with different DPSS capabilities, we believe that such a binary distinction does not represent today's product landscape adequately. Instead, the capabilities built into today's reader are quite diverse. On the one end of the spectrum, there are readers that focus on providing the base service set (BSS). On the other hand, there are readers that allow for custom code to be executed on the readers itself, which allows the readers to operate autonomously [9]–[11]. The majority of RFID readers currently available provide some limited data processing services (DPSS), such as pre-defined filters and aggregates over the RFID data captured. This includes the post-capture elimination of redundant reads and the accumulation of tag reporting across antennas [12]. Reader also compute entry/exit aggregates [13]. Some RFID reader products also provide limited persistent storage space so no data are lost during a communication failure with the backend IT systems. RFID readers with significant computing resources execute application code on the reader platform [9]–[11]. The result is that the RFID reader can control independently all local interaction, e.g. with sensors and displays. The reader only transmits the application dependent high-level events that result from the data processing, e.g. verifies a shipment against an advance shipping notice and sends a 'shipment complete' event to the enterprise resource planning system.

From a configuration service perspective (CSS), most RFID readers allow users to configure RF transmitter settings, network interface, and antenna and tag selection parameters (cf. Table II). Base service scheduling without network access, such as the immediate writing to memory upon seeing a particular tag ID, was typically only found on the RFID readers that allow users to run custom application code on the reader. However, since this feature is part of the recently released EPCglobal LLRP protocol, it will also be available on readers that support LLRP, but do not provide a local runtime environment. Access to air protocol specific settings such as timing and modulation parameters is often restricted, while application-dependent settings such as dense-reader mode are usually exposed. Most RFID readers allow systems that host monitoring services (MSS) to check network communication, e.g. via heartbeat messages. Frequently, there is also the possibility to monitor noise levels to measure RF interference. Readers can also monitor each other's health and adjust configurations in a peer-to-peer fashion. This is not a common practice today, but has been demonstrated at trade shows. The configuration and monitoring services are thus almost exclusively carried out by an additional device.

V. RFID SYSTEM ARCHITECTURE TAXONOMY

In this section, we present different RFID system architectures that are currently deployed throughout industry. The analysis is based on interviews with companies which installed RFID systems. Based on our analysis, we distinguish two different architecture types: An autonomous and a centralized architecture (cf. Fig. 5 and 6). In practice, there are also a number of hybrid architectures that feature elements of both architecture types. The deployment diagrams that illustrate these architecture types have a number of different dimensions (cf. Fig. 5). Each box in the de-



(a) Centralized with controller device

(b) Centralized without controller device

Fig. 5. Centralized Architecture (with and without controller)

ployment diagram represents a separate device. Each device has a number of services associated which are grouped into base, configuration, monitoring, and data processing services. The deployment diagrams also shows the communication link type (LAN/WAN) between different devices and group devices which are hosted within the same facility, e.g. a store or distribution center.

Fig. 5 shows an architecture with a dedicated controlling device at each facility, where one or more RFID readers are deployed. We call this the centralized architecture because a (local) central device provides CSS, MSS and the majority of the DPSS services. Existing enterprise IT monitoring systems do not monitor reader devices directly. Fig. 5 presents two variations of this architecture type. Fig. 5(a) features a separate application server[1] and a controller. The controller provides the CSS and MSS services and application-agnostic DPSS services such as persistent storage of tag reads, identifier translation and aggregation across multiple readers. The application server hosts customized application software that process the captured RFID data in a business context. In Fig. 5(b), these data processing services are deployed on the same device. In both cases, the RFID readers provide base services, such as tag identification and memory access, and limited data processing services. The latter includes the elimination of redundant reads by computing entry/exit events and the aggregation of tag reads across different antennas.

In the autonomous architecture, there is no local controller, but the readers operate autonomously once configured appropriately (cf. Fig. 6). Extensive data processing takes place on the RFID readers themselves, where custom application code processes the captured tag data. The readers send locally computed business events such as ‘shipment complete’ to the enterprise information systems. Before the captured RFID are processed in a business context, the data are typically filtered and aggregated and tag identifiers are translated. To deal with network and system failures, there is the need to provide reliable messaging and persistent storage services. The readers are monitored and also configured via enterprise system and network management tools. Each of the two architecture types has its own strengths and weaknesses with respect to performance, ease of maintenance and cost. The most suitable system architecture is thus dependent on the specific application and enterprise IT organization. Installations with hundreds of readers in the same facility typically favor a centralized architecture, while the deployment of isolated readers in remote locations or in applications with significant local interactions with staff benefit from

[1]The term ‘Application Server’ refers here to a server that provides a runtime environment for the application. While the term ‘application server’ is often associated with enterprise-class servers that implement the Java Enterprise Edition or Microsoft .NET framework, application servers hosted on-site in RFID systems feature often only a more lightweight execution environment. In an RFID context, these servers are often also referred to as ‘Edge Servers’.

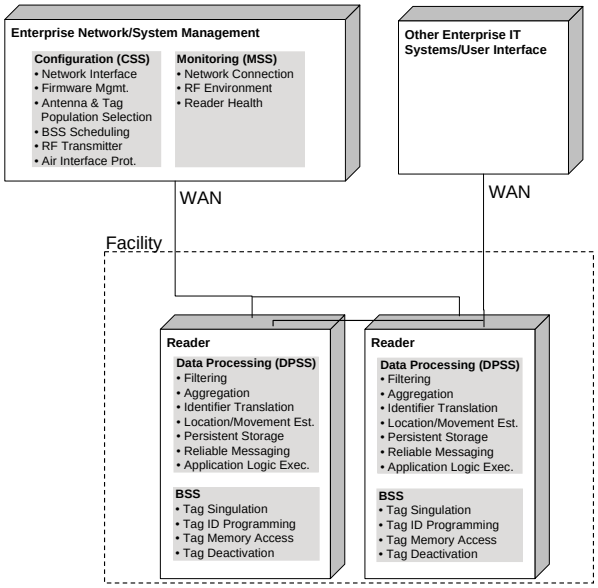


Fig. 6. Autonomous Architecture.

the autonomous architecture. In practice, there are many hybrids of the architectures types presented here deployed. This includes autonomous architectures for readers in remote locations and centralized architectures in facilities with large number of RFID readers deployed by the same organization. There are also cases where the RFID readers are monitored remotely, but the data processing is carried out on a local server/controller.

There are a number of similarities between these architecture types and the ones commonly found in WLAN access point deployments. Yang et al. distinguished three different WLAN architectures in his analysis [6]. There is an autonomous architecture where each wireless termination point is an autonomous, physical device that implements all 802.11 services. These 'standalone' access points are configured and monitored via existing network management systems. Yang et al. also identified a centralized architecture where a centralized controller (commonly referred to as an access controller) that controls and monitors the wireless termination points but also acts as a bridge and router. Yang et al. also describe a third type of architecture "in which the participating wireless nodes are capable of forming a distributed network among themselves, via wired or wireless media" [6]. RFID system architectures are similar to the WLAN architectures in that there is also the need to control the access to a shared wireless medium, the devices need to monitored, and possibly updated. RFID deployments differ from WLAN deployments however when it comes to the data processing services which play no role in WLAN deployments. WLAN deployments are application agnostic and route messages from wireless clients. Control is required to deal with node mobility and network security. There is no need to interpret data captured over the wireless interface and generate the appropriate business events as in RFID architectures.

		DCI	RM	LLRP	RP	ALE
CSS	Network Interface Configuration	■				
	Firmware Management	■				
	Antenna & Tag Population Selection			■	■	■
	Base Service Set Scheduling			■	■	■
	RF Transmitter Configuration			■		
	Air Interface Prot. Configuration			■		
MSS	Network Connection Monitoring		■	■		
	RF Environment Monitoring		■	■		
	Reader Monitoring		■	■		
DPSS	Aggregation			■	■	■
	Filtering				■	■
	Identifier Translation				■	■
	Reliable Messaging				■	■
	Persistent Storage					
	Location Movement/Estimation					
	Application Logic Execution					

Fig. 8. High-level Overview of the Services Supported By the EPCglobal Specifications DCI, RM, LLRP, RP and ALE.

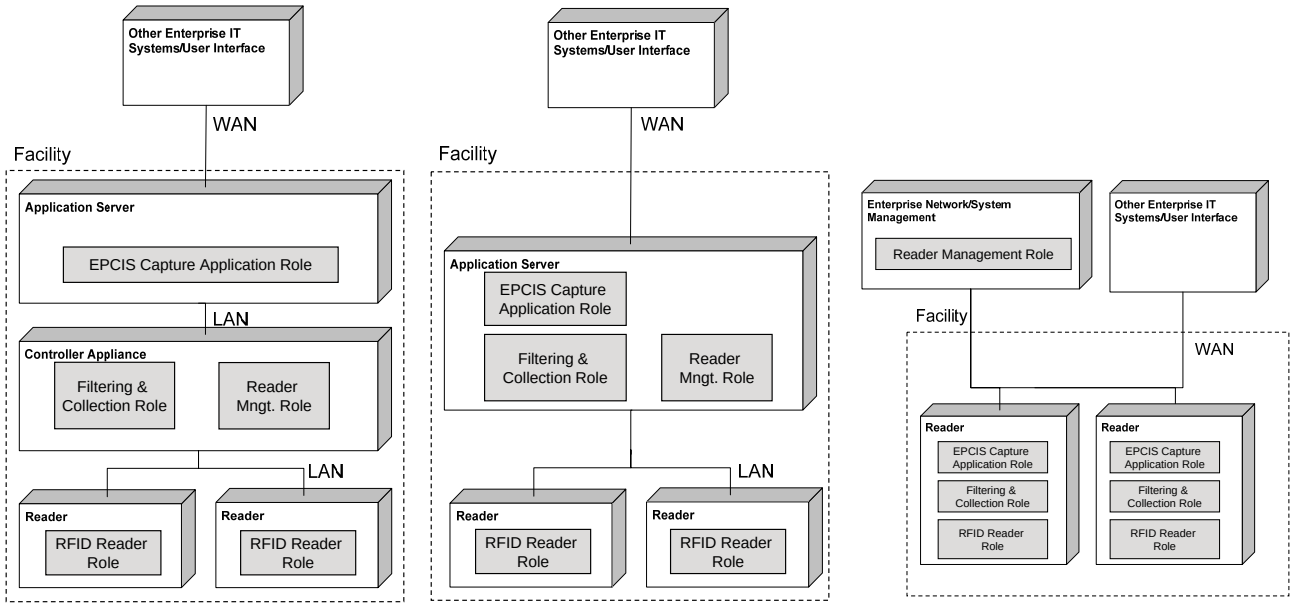
VI. EPCglobal Standards Relation to these Architecture Types and Services

The *EPCglobal Architecture Framework* defines a number of roles: *RFID reader*, *filtering&collection*, *reader management* and *EPCIS capture applications* [5]. In the 'centralized with controller' architecture we presented above, each of the roles maps to one individual device in the system architecture except for the controller that implements both the *filtering&collection* and *reader management* role (cf. Fig. 7(a)). In the variation of the centralized architecture without a separate controller device (cf. Fig. 7(b)), the *filtering&collection*, *reader management* and *EPCIS capture application* role are combined on the same device. In the autonomous architecture, the *RFID reader*, *filtering&collection* and *EPCIS capture applications* roles of the EPCglobal Architecture Framework are implemented on the RFID reader (cf. Fig. 7(c)). Each of the different roles in the EPCglobal architecture framework is associated with one or more software specifications that standardize interfaces [14]–[20]. In the following subsections, we discuss each of these specifications in the context of the service taxonomy presented earlier.

A. Discovery, Configuration and Initialization (DCI) and Reader Management (RM)

The EPCglobal DCI specification [19] supports the CSS services of network interface configuration and firmware management (cf. Fig. 8). The EPCglobal RM protocol [17] supports MSS services[2] (cf. Fig. 8). RM does not expose any air interface protocol specific statistics, but reports noise levels, transmit powers and failed tag memory access and deactivation operations. RM supports the SNMP protocol and thus facilitates the integration of RFID readers into existing enterprise system monitoring tools which rely on the SNMP protocol.

[2]In addition to the support for monitoring services, the EPCglobal RM Specification also allows hosts to selectively switch off reader antennas via the *ReadPoint.setAdminStatus* method. Strictly speaking, the latter represents a configuration service according to our service taxonomy.



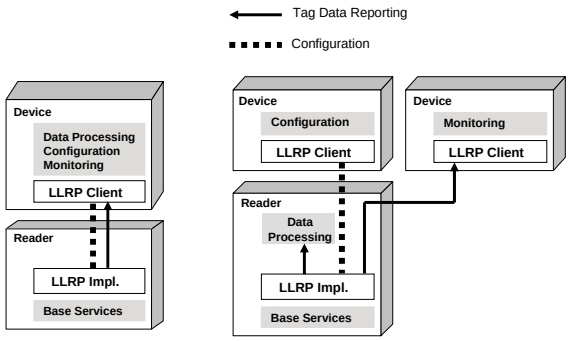
(a) Centralized with Controller (b) Centralized without Controller (c) Autonomous

Fig. 7. *Centralized and Autonomous Architectures with associated EPCglobal Architecture Framework Roles [5].*

B. Low Level Reader Protocol (LLRP)

In the recently ratified LLRP specification [18], there is extensive support for the CSS and MSS services listed in Table II and IV. This includes the configuration of air interface parameters, RF transmitter settings, base service set scheduling, and antenna, tag population and tag memory selection (cf. Fig. 8). The LLRP specification also provides limited data processing capabilities, such as aggregation across reader antennas. The *Accumulation of TagReportData* in the LLRP specification allows for the collection of tag data across multiple reader antennas. The elimination of redundant reads is possible in individual reporting cycles. The computation of entry and exit event aggregates is not supported. The latter feature would eliminate the dissemination of tag data reports until a new tag ID is detected. There is no support for the software filtering of tag IDs, but clients can specify the target tag populations that gets identified over the air interface using the 'Select' command in ISO 18000-6C.

LLRP is well-suited to standardize the reader interface in the centralized architectures. LLRP allows controllers and software deployed on local application servers to schedule base services and to optimize performance by adjusting RF transmitter settings and air interface parameters. The LLRP specification envisions only a single connection to a client at a time (cf. Fig. 9(a)). This allows for one device configuring the reader and subsequently for another device to connect to the reader to receive and process the *TagReportData* notifications. It does not permit sending the *TagReportData* to multiple data consumers (cf. Fig. 9(b)). LLRP by itself can thus not be used to support a deployment scenario where applications on the reader process the data captured and remote



(a) Single channel (b) Multiple reporting channels

Fig. 9. Simultaneous notifications to multiple data consumer are not supported by today's LLRP specification.

system monitoring services receive notifications to monitor the health of the reader and connection. However, LLRP can be operated in conjunction with RM. This provides SNMP access to the reader status and statistics on successful and failed base service operations for remote system monitors.

LLRP can also generate significant network traffic since there is no way to restrict *TagReportData* messages to tags entering and exiting the range of the reader when low notification latencies are desired. The latter is not an issue for local area connections, but can be problematic for wide area connections. The binary communication protocol allows for efficient implementations on RFID readers, but requires additional high-level tools for application developers.

LLRP features an extension mechanism that allows reader vendors to define their own proprietary extensions that expose custom feature sets. In our opinion, this is essential

since RFID reader vendors will only consider substituting their proprietary reader protocols for a standardized protocol if the latter allows for extensions supporting proprietary features which are not part of the standard feature set. While LLRP in its current version only features support for the ISO 18000-6C air interface, the support for other air interface protocols is envisioned in future generation of the specification.

C. Reader Protocol (RP)

The EPCglobal RP specification [16] takes a different approach to interface with a reader in two important aspects when compared to LLRP. While LLRP comprises CSS, MSS and limited DPSS services in one protocol, RP focuses on DPSS functionality (and limited CSS services) with the complementary EPCglobal specification RM supporting MSS services (cf. Fig. 8). The second major difference is that the combination of RP and RM does not allow CSS services access to RF parameters and air interface protocol settings of the reader[3] – the protocols are often referred to as 'air interface protocol unaware'. This represents a deliberate design choice. The benefit of this approach is that application developers are shielded from the details of RFID operation. Software application development is facilitated by providing a standardized high-level RFID data reporting interface. Another benefit of this approach is that there is no need to reach consensus on the configuration and control features to support in a standardized protocol and possibly across different air interface protocols. An HF reader vendor can for example continue to use his own proprietary reader protocol to configure its readers, but the tag data reporting takes place via a 'high-level' reader protocol and is identical to the tag data reporting of UHF Gen2 or ISO 18000-6B readers also using RP to report captured data.

The drawback of this design choice is that the reader cannot be controlled and configured to the extent this is possible with today's proprietary reader protocols or with the recently released LLRP. It is for example not possible to select a particular air interface protocol or an interference avoidance features such as dense reader mode remotely. It is also not possible to select a particular frequency channel to minimize reader collisions. All of these represent standard features offered by today's vendor specific reader protocols. Since access to these features is necessary at least during the initial deployment, RP requires the use of another vendor specific protocol to configure the reader appropriately (cf. Fig. 10). In some applications, the configuration via another 'air interface aware' (proprietary) protocol is only required at the time of deployment with no changes during operation. In other applications, RF interference or changes in tag populations for example require frequent changes to the reader configuration to maintain optimum performance. This results in frequent use of the additional 'air interface aware' (proprietary) protocol.

RP supports DPSS services that are not included in the LLRP specification. RP allows for the computation of entry and exit events, different tag ID representations and tag report notifications to one or more data consumers. RP also provides an XML message binding that facilitates software application development, while LLRP relies on a binary protocol that requires fewer resources on RFID readers. RP supports reliable messaging for the asynchronous notification channels.

Nearly all of the DPSS services in RP are optional in the specification. This design choice seems to be a direct result of the heterogeneity of the reader landscape, where reader devices with significant computing resources were envisioned to provide the majority of the optional features and low-end reader devices would only support the mandatory features and possibly a small number of the optional features. However, this design choice also makes application development with RP a challenge because different reader types will likely support a different combination of optional features.

D. Application Level Event (ALE)

The EPCglobal ALE 1.0/1.1 [14], [15] specifications provide a standardized interface to application-agnostic DPSS services. ALE also supports the selection of readers, reader antennas and tag populations, which represents a CSS service in our taxonomy. ALE 1.0 comprises a feature set similar to the EPCglobal RP specification: Filtering, aggregation, tag identifier translation, buffering of messages and notifications to multiple consumers, but the actual implementation is different. ALE relies for example on a web service (SOAP) transport protocol and provides a convenient 'subscribe' mechanism where clients can register their standing queries/notifications with a single command. In RP, this requires a sequence of individual messages. ALE 1.0 does not support tag user memory.

ALE 1.1 represents a major advance over ALE 1.0. ALE 1.1 supports reading and writing to user memory on

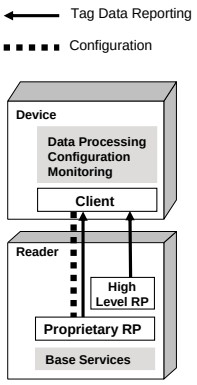


Fig. 10. High-level reader protocols such as EPCglobal RP and ALE require an additional (proprietary) protocol to configure the reader.

[3]RP only allows for the CSS services of antenna and tag population selection and limited base service scheduling. Via RP, clients can schedule inventory ID operations, e.g. with timers, but memory access operations cannot be scheduled and need to be triggered remotely.

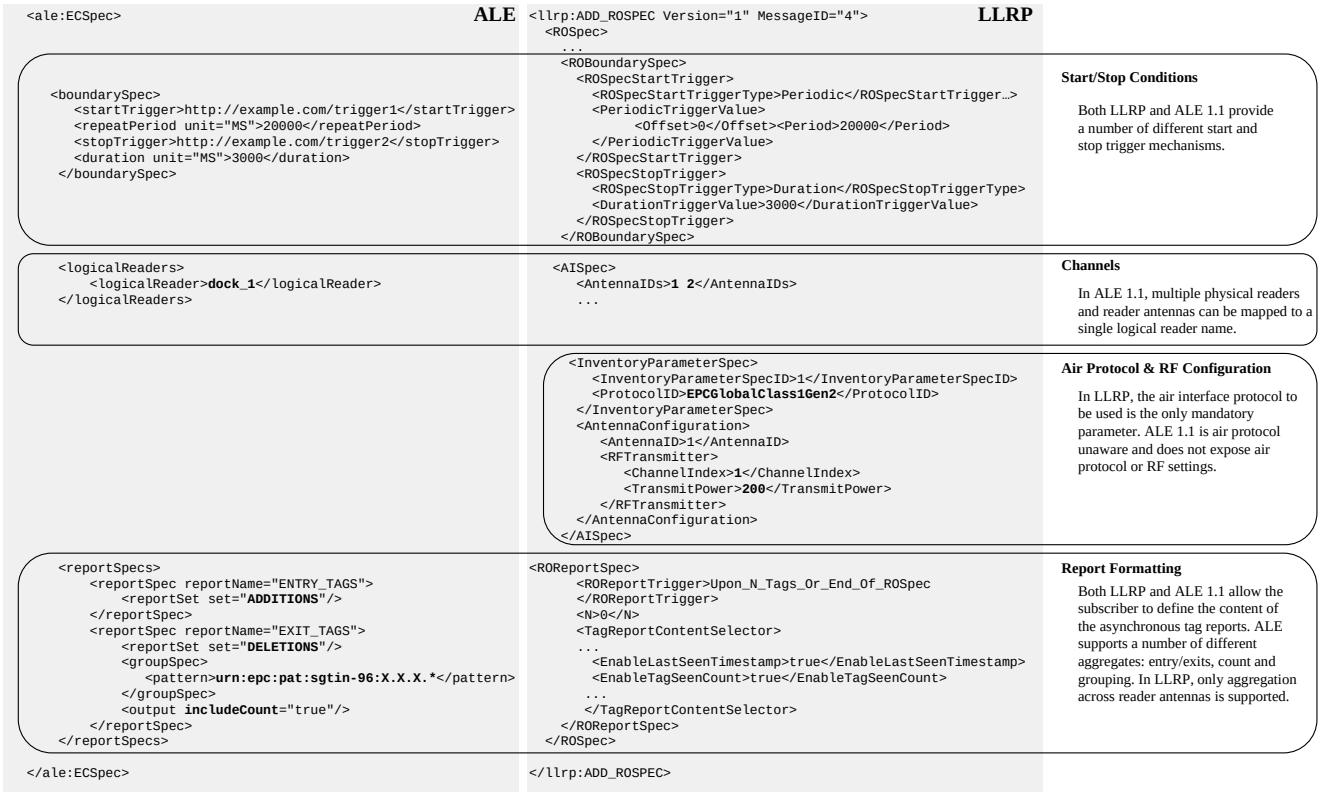


Fig. 11. Specification of tag identification in ALE 1.1 and LLRP. The LLRP *ADD_ROSPEC* message is represented in the LLRP LTK XML format (www.llrp.org).

tags. ALE 1.1 also provides new interfaces for defining tag memory fields, for reader/reader antenna to location mapping and for access control.

In LLRP and ALE 1.1, the primary interaction between client and implementation for both tag identification and memory access are similar. For tag identification, the ALE/LLRP client provides the implementation with a specification (*ECSpec* in ALE/*ROSpec* in LLRP) that defines boundary conditions, channels to be used and the desired content and structure of the asynchronous reports (cf. Fig. 11). The implementation executes this specification, captures the RFID data and responds by returning the information in the reports as requested (cf. Fig. 12).

For the tag memory access, the primary interaction sequence is also similar for ALE 1.1 and LLRP. The client transmits a specification (*ECSpec for reading/CCSpec for writing* in ALE/*AccessSpec* in LLRP) (cf. Fig. 13). The ALE/LLRP implementations respond by carrying out the memory access operations on the tags and return reports that describe which memory access operations were performed.

While the basic interaction is similar, there are a number of conceptual differences between the LLRP and ALE 1.1 specification. LLRP allows the client to optionally specify air interface protocol and RF transmitter settings in *ROSpec* and *AccessSpec* (cf. Fig. 11). The only mandatory air interface protocol parameter is the air protocol to be used, e.g. ISO 18000-6C. The ALE specification abstracts from these low level settings and does not expose them. The ALE 1.1 specification uses 'high-level' representations for tag identifiers, such as the URI representations of EPCs defined in the EPCglobal Tag Data Standard [21]. In LLRP, EPCs are represented as bit arrays (cf. Fig. 12). ALE 1.1 includes the 'logical reader API' which decouples the identity of reader devices and antennas from the names of the channels used in ALE subscriptions and reports. This permits the replacement of an RFID reader or a change of networking parameters without the need to update the application software. As mentioned earlier, ALE also provides additional data processing services, such as count and entry/exit aggregates, which are not available in LLRP (cf. Fig. 12).

For tag user memory, ALE 1.1 provides predefined memory field names for elements specified in the EPCglobal Tag Data Standard [21] and support for ISO 15962. Both of which facilitate the programming of memory access operations. In LLRP, tag user memory needs to be addressed using memory banks, pointers and length. Masks need to be specified as bit arrays (cf. Fig. 13). An ALE implementation can also service multiple data consumers simultaneously and disseminate captured tag data to these. LLRP is a network protocol that supports a single established connection at a time only and does not support multiple clients with different 'subscriptions'.

The main use case of the EPCglobal ALE specification is on controllers and application servers in the centralized

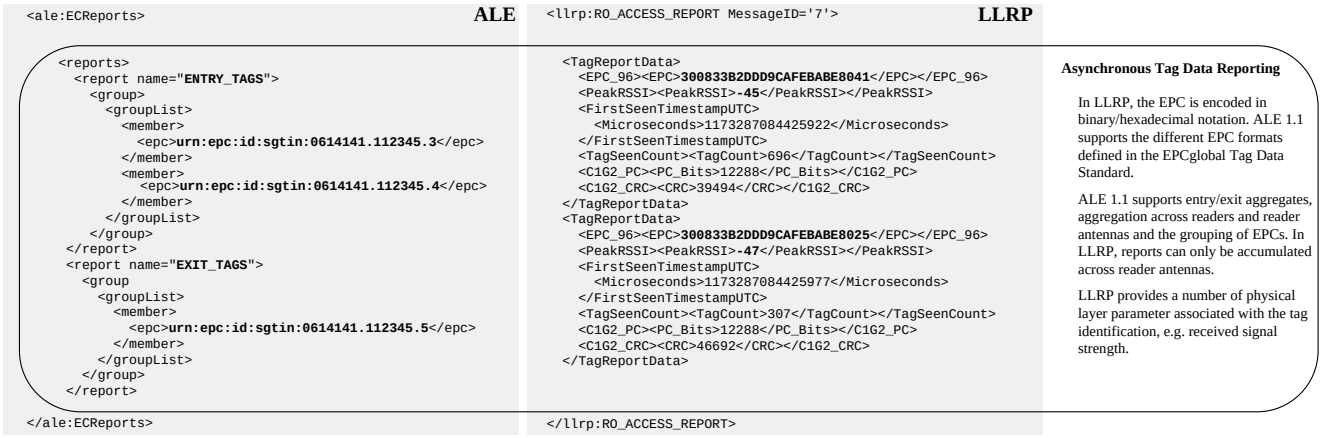


Fig. 12. Tag data reports in ALE 1.1 and LLRP. The LLRP *RO_ACCESS_REPORT* message is represented in the LLRP LTK XML format.

Fig. 13. Tag memory access (reading) in ALE 1.1 and LLRP. The LLRP *ADD_ACCESSSPEC* message is represented in the LLRP LTK XML format..

architectures as an application agnostic interface (cf. Fig. 14), where ALE exposes all those DPSS services that are common across different RFID applications, e.g. filtering, aggregation and tag identifier translation. The ALE interface also allows clients to select readers, reader antennas and tag populations and define tag memory operations. ALE does not provide access to RF transmitter and air protocol settings. At the interface to software and hardware controllers, the control and data plane can often be separated and it is appropriate to abstract from the RFID operational settings. There is no need to expose air protocol settings because the controller implementation is responsible for RFID performance optimization.

ALE can also be deployed on an RFID reader as a 'high-level' reader protocol with the similar benefits and weaknesses as mentioned in the subsection on RP. ALE provides a high-level starting point that is well-suited for writing application logic, freeing the developer from the kind of low-level programming that would be necessary to code directly to LLRP. The application developer benefits in particular from the high-level tag identifier and user memory representation in ALE 1.1. In LLRP, these are represented in binary notation. The drawback of ALE as a high-level reader protocol is the need for an additional (proprietary) protocol that configures the reader, since the configuration of the RF transmitter and air protocol settings is at least required during the initial deployment (cf. Fig. 10).

E. EPC Information Service (EPCIS)

EPCglobal's EPCIS specification [20] deals not just with raw RFID observations, but defines events that link the raw observations reported by LLRP, ALE, or RP with meaning relative to specific steps in business processes. EPCIS-level events stored in an EPCIS repository are thus a result of combining RFID data captured with knowledge about the significance of an 'RFID read'. Due the diverse nature of business processes, the actual data processing that leads to the generation of EPCIS-level events typically relies on

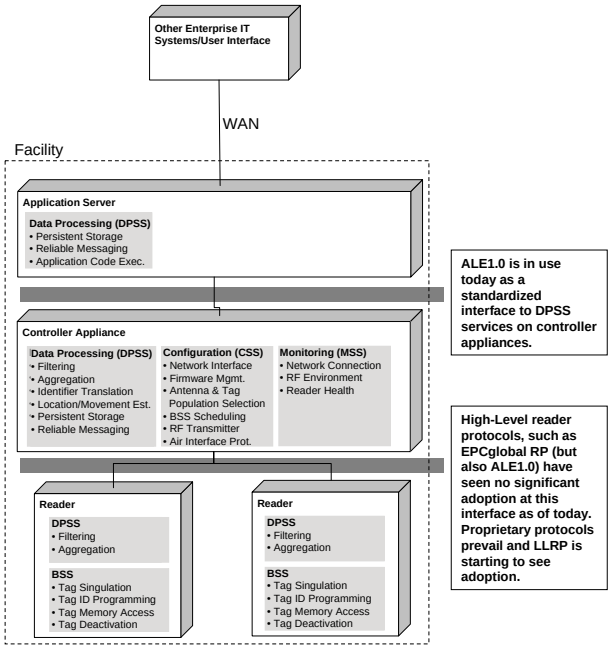


Fig. 14. Adoption of EPCglobal specifications ALE, RP and LLRP in the centralized architecture.

custom application code. In the centralized architecture, such 'capture applications' are deployed on application servers. In the autonomous architecture, the corresponding application code is executed on the reader itself. The reader transmits the EPCIS-level events to a remote EPCIS repository using the 'capture' interface of the EPCIS specification. The EPCIS repository is then accessed by other enterprise applications or trading partners via the EPCIS 'query interface'. EPCIS deals explicitly with historical data stored in a persistent data repository. In contrast, LLRP, RP and ALE of are oriented exclusively towards real-time processing of EPC data.

In the service taxonomy defined in this paper, the EPCIS specification thus supports data processing services that execute custom application logic by defining the event types generated by these applications. The EPCIS specification also provides the interfaces to persistent storage for EPCIS-level events.

F. Tag Data Translation Specification (TDT)

The TDT specification defines the current Tag Data Standard encoding and decoding rules in an unambiguous machine-readable format. This facilitates the translations between different representations of electronic product codes by any software component in an RFID deployment. For example, it could translate from the binary format for a GTIN on a 96-bit tag to a pure-identity URI representation of the same identifier. In our service taxonomy, the machine-readable translation rules of the TDT specification support the tag identifier translation service.

VII. Adoption of High-Level Reader Protocols

The majority of the standards listed above have been released within the last twelve month which makes it difficult to assess adoption at this stage. However, ALE 1.0 and RP have been available since 2005 and early 2006, respectively. As mentioned earlier, ALE 1.0's main use case is today as a standardized interface between application logic and controller devices (cf. Fig. 14) or software controllers on application servers. ALE 1.0 implementations have been certified by more than a dozen vendors of hardware and software controllers for use in the centralized architecture. Since the EPCglobal RP specification provides a similar feature set to ALE 1.0, the RP specification is also applicable as an interface to a controller. However, RP sees in practice hardly any usage as an API on software and hardware controllers.

RP and also ALE 1.0/1.1 have been proposed as high-level reader protocols for the interface to readers. However, in this role RP and ALE 1.0 have seen very limited adoption. We are only aware of a single reader vendor offering an EPCglobal RP compliant reader [22] and a single reader vendor offering an EPCglobal ALE 1.0 compliant reader [10]. Nearly all RFID system deployments today rely on proprietary reader vendor specific reader protocols (cf. Fig. 14). In our opinion, the lack of adoption of 'high-level' reader protocols has a number of reasons:

- **Focus on UHF Gen2 by reader vendors.** The development of these high-level reader protocol standards coincided with the ratification of the air interface protocol Gen2. This led reader vendors focus on the development of new UHF Gen2 reader products. The result was that the development of the specifications took place with very limited participation from reader vendors.
- **Limited computing resources on first generation UHF readers.** High-level reader protocols require significant computing resources which were not always available on first generation UHF readers. The demand for significant computing resources result from the use of XML messaging, asynchronous notifications to multiple data consumers and buffering, aggregation and filtering on the reader.
- **Uncertainty about the 'right' protocol to support.** The availability of two specifications with nearly identical features, RP and ALE 1.0, but very different nomenclature and implementations led to some confusion about the 'right' protocol to support. The ongoing development of yet another reader protocol, LLRP, added to the uncertainty.
- **Limited functionality in high level reader protocols.** ALE 1.0 did not support tag user memory, which was added in the recently ratified ALE 1.1.
- **Limited demand by end users and RFID 'middleware' companies.** Neither end users nor RFID 'middleware' companies requested reader vendors to support any of the two high-level reader protocols. In interviews with a small set of end-users we conducted, no company representative indicated that neither RP nor ALE 1.0 addressed their needs as reader protocols. The end user companies we interviewed disliked the

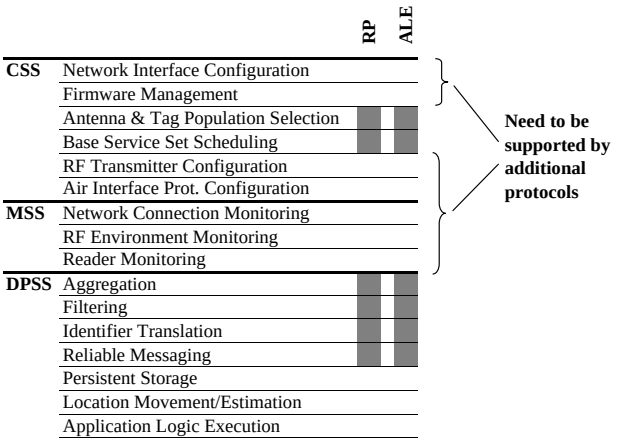


Fig. 15. High-level reader protocols such as ALE or RP require additional protocols for configuration and monitoring purposes.

idea of high-level reader protocols and requested access to RF transmitter and air protocol settings. End user companies mentioned that such high-level interfaces are more appropriate as an interface to software and hardware controllers in the centralized architecture.

- **High-level reader protocols are not a replacement for existing proprietary reader protocols.** Since reader vendors need to continue offering their proprietary protocols to configure and monitor their readers in addition to the high-level reader protocols for reporting aggregated and filtered RFID data, the support of a high-level reader protocol meant the de-facto support of two reader protocols (cf. Fig. 15).

Major RFID reader vendors we contacted during this investigation mentioned that they have no intention to support RP or ALE on their readers in the short term. Instead, there seems to be a trend to support the LLRP protocol instead [10], [23], [24]. Similarly, end user companies mentioned that they would not request the support of RP or ALE by reader vendors in the near future.

It remains to be seen whether the recently ratified specification ALE 1.1 will see more adoption on reader devices than RP and ALE 1.0 in the past. ALE 1.1 addresses the shortcomings of ALE 1.0 and provides a convenient high-level interface for tag user memory operations. Computing resources on today's readers also increased. The above analysis however indicates that high-level reader protocols will be adopted as interfaces to reader devices only if they are augmented by an additional protocol that supports fine-grained configuration services – effectively dismissing the idea of a 'high-level only' reader protocol as the sole interface to a reader device. ALE 1.1 will thus also require an additional (proprietary) protocol to configure and monitor the reader (cf. Fig. 15). As noted above, high-level interfaces such as ALE do have an important role to play at a higher level in the software stack.

VIII. Potential Future Standardization Activities

There are a number of potential future standardization activities. This includes enhancements of existing standards but also the standardization of new interfaces. In our opinion, one should consider enhancing the LLRP specification with a few additional optional features:

- **Entry/Exit aggregates.** In our previous description of LLRP, we already proposed to add optional support for entry/exit aggregates. We believe that this would reduce the network traffic which is important for remotely deployed readers.
- **Filtering.** LLRP currently only allows selecting a tag population over the air interface. Optional filtering in software would help to eliminate those tag reads that resulted from RFID tags missing the Gen2 *Select* command.
- **XML representation of the LLRP tag data reports.** LLRP uses a binary messaging protocol. To facilitate RFID application development, one could consider standardizing an XML interface for the LLRP tag data reports. Fig. 12 shows such a high-level representation that was developed in the *LLRP Toolkit Project*[4]. Having an XML interface for the asynchronous notification channel only would not require an XML parser on the reader.
- **Multiple tag data reporting channels.** We propose to optionally allow for more than a single connection at a time. This would allow readers to send captured data to more than one event consumer (cf. Fig. 9). The introduction of optional multiple tag data reporting channels would allow for the separation of monitoring and data plane in applications where this is appropriate without requiring the use of RM.

In those cases, where RM and LLRP are implemented on the same reader device, we also propose to specify a common approach to deal with the 'monitored objects' in RM that are specific to RP, e.g. 'Sources' and 'NotificationChannels'. Otherwise, different implementations of RM only (without RP) are likely to deal with these 'monitored objects' in different ways. One might also consider making RM 'air protocol aware' by exposing some of the RF transmitter settings and air protocol parameters via the SMNP interface of the RM specification.

Future standardization efforts might also want to consider specifying the runtime environment on RFID readers to support autonomous architectures (cf. Fig. 16). There are already many RFID reader products on the market that allow users to execute application code on the reader. Agreement on a common virtual machine on these devices and a mechanism to upload applications would greatly facilitate the application development. The development of custom application logic on the reader would also benefit from a standardized internal interface to receive captured tag data and to configure the

[4]www.llrp.org

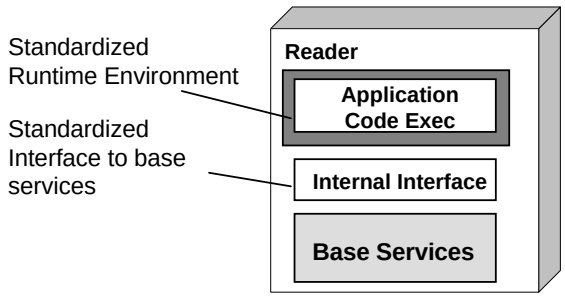


Fig. 16. RFID reader with standard runtime environment for application code and internal interface to reader module providing the base services such as tag inventory and memory access.

reader base services. This could be realized with an LLRP implementation that supports multiple reporting channels (cf. Fig. 9). Alternatively, the ALE 1.1 interface with its high-level representation of tag identifiers and user memory is also a suitable candidate. It would need to be augmented by an additional interface to configure and monitor the reader module.

IX. Conclusion

As the use of RFID systems increases, large scale deployments of RFID are highlighting certain challenges. The management and control of large number of RFID readers is becoming an issue from a network administration and data management perspective. In this paper, we categorize the services that typically comprise an RFID system into base, configuration, monitoring and data processing services. We use this service taxonomy to distinguish different system architectures deployed in the industry today. While we do not believe that the list of services we provided is necessarily exhaustive, we believe that they provide an adequate framework to analyze RFID deployments.

In our analysis, we identify two main types of system architectures. We distinguish a centralized architecture with a local hardware or software controller managing the RFID readers which provides configuration, monitoring, and data processing services and an architecture where the reader operates independently, has significant data processing capabilities itself and is configured and monitored remotely. There also exist a number of hybrid architectures that combine elements of the centralized and autonomous architecture type.

The EPCglobal community has developed a set of specifications that aim to standardize the interfaces to the configuration, monitoring and data processing services. In our opinion, the combination of DCI and LLRP specification is well-suited to standardize the communication interface between RFID readers and local hardware and software controllers in centralized architectures. The EPCglobal ALE specification is today already in use in the centralized architecture on a controller device or an application server where it provides a convenient interface for data processing services across RFID readers that are application agnostic, such as aggregation and filtering. Our analysis also discusses a number of potential enhancements to the LLRP specification.

This paper also discusses 'high-level' reader protocols such as EPCglobal RP and ALE that abstract from the underlying air interface protocol and RF details. These 'high-level' reader protocols shield the application developer from low-level details of RFID and provide a convenient starting point for the development of application logic. As of today, neither EPCglobal RP nor ALE 1.0 have seen significant adoption by reader vendors. We argue that the main reason for this lack of adoption are the need to supplement them with an additional protocol for configuration and monitoring purposes and the lack of demand from end users and RFID middleware vendors. It remains to be seen whether the recently ratified ALE 1.1 with its enhanced feature set will lead to increased demand and adoption as a high-level reader protocol on reader devices. As noted above, high-level interfaces such as ALE1.1 play an important role at a higher level in the software stack.

Future standardization efforts might want to consider specifying the runtime environment on RFID readers to support autonomous architectures. Today application code to be executed directly on a reader device needs to be customized for each reader platform. A common runtime environment and upload mechanism and a standardized interface to the reader module would greatly facilitate application development in such an autonomous architecture. Similar to today's mobile phones, future RFID readers might thus feature a standardized virtual machine supporting an RFID reader device profile.

References

[1] C. Bornhövd, T. Lin, S. Haller, and J. Schaper, "Integrating Automatic Data Acquisition with Business Processes - Experiences with SAP's Auto-ID Infrastructure," in *Proceedings of the 30st international conference on very large data bases (VLDB)*, (Toronto, Canada), pp. 1182–1188, VLDB Endowment, 2004.

[2] C. Floerkemeier, C. Roduner, and M. Lampe, "RFID Application Development with the Accada Middleware Platform," *IEEE Systems Journal: Special Issue RFID*, Jan. 2008.

[3] S. Sarma, "Integrating RFID," *ACM Queue*, vol. 2, no. 7, pp. 50–57, 2004.

[4] F. Wang and P. Liu, "Temporal management of RFID data," in *Proceedings of the 31st international conference on very large data bases (VLDB)*, pp. 1128–1139, VLDB Endowment, 2005.

[5] Architecture Review Committee, "The EPCglobal Architecture Framework." EPCglobal, Sept. 2007.

[6] L. Yang, P. Zerfos, and E. Sadot, "Architecture Taxonomy for Control and Provisioning of Wireless Access Points." RFC4118, June 2005.

[7] EPCglobal, "Class 1 Generation 2 UHF Air Interface Protocol Standard Version 1.0.9," 2005.

[8] International Organization for Standardization, "Information technology – Radio frequency identification for item management – Part 6: Parameters for air interface communications at 860 MHz to 960 MHz ," 2004.

[9] Impinj, "Speedway Product Specification." www.impinj.com.

[10] Intermec, "IF61 Enterprise Reader Product Profile." www.intermec.com.

[11] Thingmagic, "MercuryOS API Reference Guide." www.thingmagic.com.

[12] Impinj, "Mach1 API Reference Guide." www.impinj.com.

[13] Alien, "Alien API Reference Guide."

[14] EPCglobal, "The Application Level Events (ALE) Specification, Version 1.0." www.epcglobalinc.org, Sept. 2005.

[15] EPCglobal, “The Application Level Events (ALE) Specification, Version 1.1.” www.epcglobalinc.org, Sept. 2008.
[16] EPCglobal, “Reader Protocol Standard, Version 1.1.” www.epcglobalinc.org, June 2006.
[17] EPCglobal, “Reader Management 1.0.1.” www.epcglobalinc.org, May 2007.
[18] EPCglobal, “Low Level Reader Protocol (LLRP), Version 1.0.1.” www.epcglobalinc.org, Aug. 2007.
[19] EPCglobal, “Discovery, Configuration and Initialization (DCI) End-User Preview.” www.epcglobalinc.org, 2007.
[20] EPCglobal, “EPC Information Services (EPCIS) Version 1.0, Specification.” EPCglobal, Apr. 2007.
[21] EPCglobal, “EPCglobal Tag Data Standards Version 1.3.1.” www.epcglobalinc.org, Mar. 2006.
[22] Elektrobit, “EB RFID Reader URP-1000 ETSI.” www.elektrobit.com.
[23] Impinj, “Impinj Revs Up Speedway Reader with Octane 3.0 Firmware for Enterprise-Ready RFID Solutions.” www.impinj.com.
[24] Intelleflex, “Datasheet FMR-5000 Reader.”