10

Accelerating Time to Value for RFID Solutions with Reusable Assets

Han Chen, Paul Chou, Sastry S. Dury and Jim A. Laredo
IBM T.J. Watson Research Center
USA

1. Introduction

Applications of RFID technologies have progressed from isolated proof-of-concept projects to integrated business enablers that provide up-to-date visibility of physical objects in the context of business processes within an enterprise and across partners. The increased scale of the projects makes the drive for Return on Investment (ROI) more important than ever; reducing the RFID solution lifecycle costs and accelerating the time-to-value have become the key challenges to broad-based RFID adoption and deployment. This calls for that solutions be built using proven architectures based on well-understood patterns and making use of the lessons learned from one project to the next. If new solutions can be built based on existing ones with reusable parts to reduce the cost of development and integration, the ROI case becomes a lesser hurdle, and situations where once RFID would have been considered for being too costly to implement now become affordable to deliver business returns.

Like any technology, the continuous innovation and identification of new applications promote asset reuse. It reaches levels of maturity where components are better understood from many aspects of their lifecycle, such as features/functions, dependencies, performance, and ease of integration. It is common to see these components evolve bottom-up from the perspective of technology maturity. Early RFID applications focused on components that process RFID read events generated from the readers. As the reader technology improved and tags became more common due to reduced price points, components were hardened and broadened to support larger event loads and additional usage patterns. Standards started to emerge, such as Application Level Events (ALE) and Low Level Reader Protocol (LLRP), developed under the auspices of EPCglobal (EPCglobal). They provide the basis for building more reusable components regardless of the choice of specific RFID reader technology. The evolution of RFID-enabled goods shipping/receiving portal serves as a good example of component maturity. Higher-level standards, such as the EPC Information Service (EPCIS) (EPCgloba), further address the needs for component interoperability, enabling supply chain level integration supported by standards-based components.

These more robust components opened the door for more business oriented solutions, built around business processes that deliver value as measured by specific key performance indicators. These solutions can handle a greater inflow and variety of events or integrate to other components in the supply chain. Some of these RFID-enabled business processes, as

Source: Sustainable Radio Frequency Identification Solutions, Book edited by: Cristina Turcu,
ISBN 978-953-7619-74-9, pp. 356, February 2010, INTECH, Croatia, downloaded from SCIYO.COM

they matured and were better understood, became common business building blocks that are applicable for a larger set of use cases, in some cases across industries. Their realization into the supporting infrastructure becomes possible as long as it is possible to choreograph those processes across the components, in some cases requiring some level of customization, as a function of the industry where they were being used. As the components get reused, it is easier to identify its applicability and integration requirements with others. It is then possible to identify the integration points and configuration parameters, also known as Points of Variability, and create a Template that captures the expertise of those that have performed the activity many times. Follow-on integrations only require the instantiation of the template and the configuration of the points of variability to take advantage of the expert experience packaged in the template, a much simpler and faster activity compared to earlier implementations that often requires more time and skills to complete. Creating templates that facilitate the use of underlying components and help realize a high-level business process ultimately accelerates the time to value of a solution.

In this chapter, we closely examine the evolution of technology, architectures, and approaches to solution building in the RFID field. We examine those components that have benefited from standards that now have become the building blocks for solutions that when combined with business processes unleash a greater business value. Furthermore, we look at the options to templatize parts of the process to accelerate the integration and reuse of these components and to ultimately deliver a greater ROI in a shorter period of time. We conclude looking forward to what is coming up next and our possible choices to solve the open problems that remain.

2. Pharmaceutical products track and trace – a running example

In this section we describe one of the most important applications of RFID for the pharmaceutical industry – the track and trace of pharmaceutical products throughout their supply chains. We use this example to illustrate the common processes across the supply chain for enabling tracking and tracing of goods, and how these processes are supported by reusable components built around open industry standards.

In addition to the common benefits associated with RFID-enabled supply chain management, such as better inventory visibility, reduced out-of-stock, and so on, pharmaceutical track and trace addresses two even more pressing issues in this industry— drug counterfeit and diversion. Many governments including European Union, Japan and several states in United States have proposed or adopted pedigree laws or regulations to address the counterfeit problem. For example, the California law requires that each prescription drug must be serialized by the manufacture at its smallest packing unit. Every transaction starting from the manufacturer must be appended to an electronic pedigree; and wholesalers and retail pharmacies cannot take possession of a product or provide it to others without validating the chain of custody. The problem of drug diversion is a result of the multi-tiered pricing structure employed by the Pharmaceutical industry. For the exact same product, lower price is used for less affluent markets while much higher price is targeted for communities which can afford to pay more.

The RFID-enabled Pharmaceutical Track and Trace application addresses both problems. It allows serialized, RFID tagged product items to be quickly identified at each transaction.

The serial identification is used to look up and update an electronic pedigree system that is built on a standard-based information service, namely EPC Information Service (EPCIS). It provides product traceability for diversion detection and also enables product tracking for recalls.

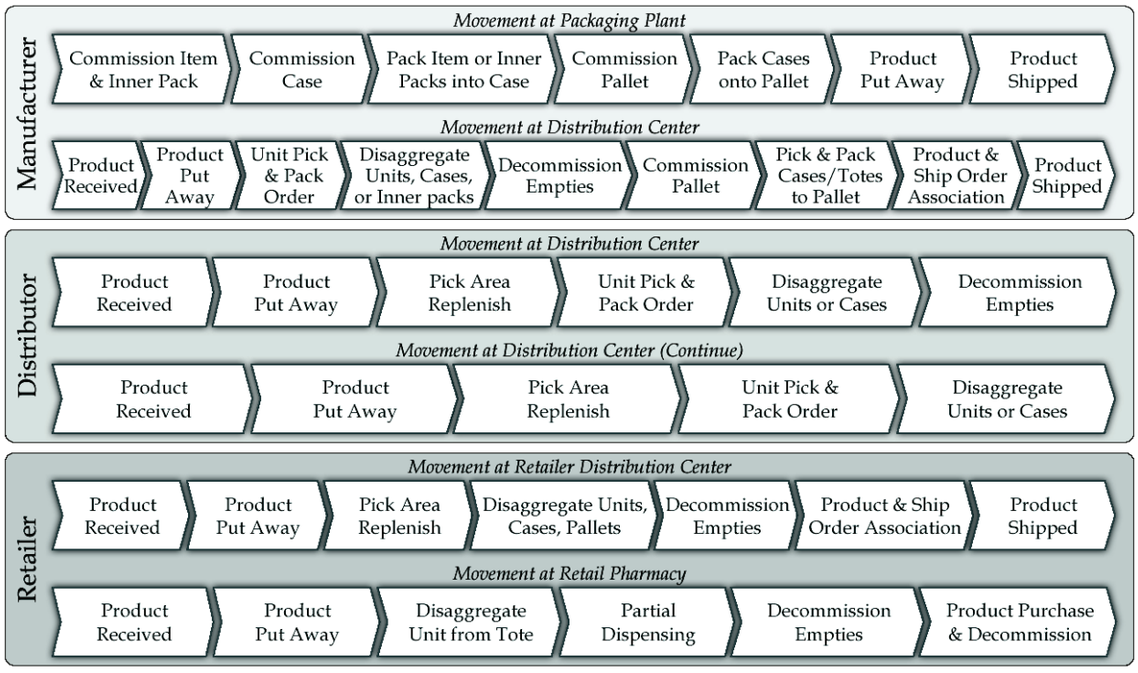

Fig. 1. Pharmaceutical Product Process Flow

Fig. 1 shows at a very high-level various steps in the pharmaceutical supply chain as drugs move from manufacturer to retail pharmacy. Many of the business process steps, such as product packing (aggregating), shipping, receiving, put-away, unpacking (disaggregating) are common, albeit with some variations, across different part of the supply chain. For example, one could define an abstract packing process to consist of the following steps: 1) determine the number of smaller units per large container, 2) fill large container, 3) verify large container contains expected number of smaller containers, and 4) create containment association in an information service.

Fig. 2 shows a simplified operational setup for an example case packing process ("Pack Items or Inner Packs into Case" in Fig. 1). The example process involves physical processing of goods wherein the pack line operator responds to sensory feedback, and initiates appropriate actions to pack case containers with a predetermined number of product units (in this case bottles).

In a typical scenario, some drug is manufactured during a production run and is filled into unit containers. The pack line operator authenticates to the system and using drug's National Drug Code, batch and lot numbers to allocate a set of unique Electronic Product Code (EPC) numbers for both unit and case containers. The pack line operator feeds this information to a label generating application to commission unit and case tags. These activities correspond to the steps "Commissioning Item and Inner Pack" and "Commissioning Case" in Fig. 1. The next step, "Pack Item or Inner Packs into Case," is described in detail below.

In the example packing process, unit containers are affixed with high-frequency (HF) tags, and cases are affixed with ultra-high-frequency (UHF) tags. Therefore, packing process needs two different RFID readers: one for reading HF tags and another for reading UHF tags. Starting from the left the bottling machine fills unit containers with specified drug. A tag slapping machine applies HF tags to bottles. A predetermined number of bottles are packed into cases. The tunnel reader reads HF tags. The UHF reader to the right of the tunnel reader reads the case tag. The light stack to the right of UHF reader shows the status of the read, specifically indicating whether the number of tags read match the number of unit containers expected in the case and whether the case tag was read successfully. If all counts match expected values, then the system creates containment association between case tag and unit tags, and saves this information in a repository.

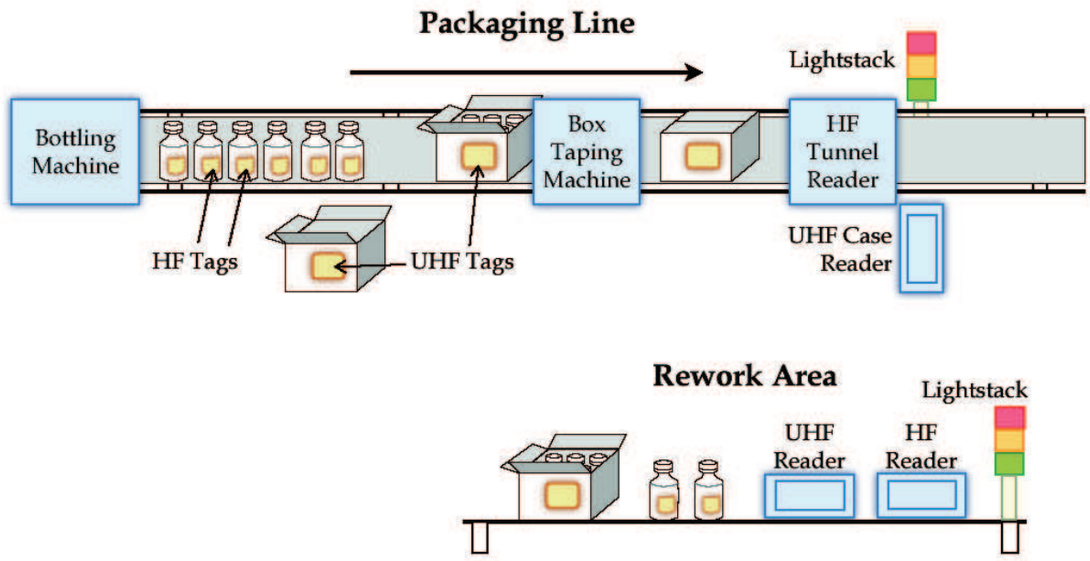

Fig. 2. Packaging Line Operational Setup

Otherwise, the conveyer is stopped and the case diverted for manual inspection later. After removing the defective case from the conveyer, the operator starts the conveyer and continues with the remaining cases. The rework area is used for the operator to manually examine/repack the cases with tag read problems as indicated by the light stack.

3. Evolution and current technologies

While RFID is still being gradually rolled out in many industries. It has reached a certain level of maturity to allow us to reflect on the evolution of its adoption and the associated software architecture. Using the pharmaceutical track and trace application as an example, we examine the trends that we have identified.

3.1 RFID technology is better understood and integrated into business process

In the early days of RFID adoption, companies have focused mainly on the technology itself. A lot of demonstration projects were conducted as proof-of-concept or technology

validation. The overriding goal of these early projects is to see if and how well the RFID technology works in specific operating environments and business processes. Read accuracy was a key focus. Due to the complex physical nature of radio transmission, many factors may adversely affect the read accuracy, for example, the operating frequency of a tag, the dielectric property of the material being tagged, the radiation power of a reader, and the relative positions of nearby readers. In the exemplar pharmaceutical packing use case described earlier, the content inside the bottle has a direct impact on the choice of tags. For non-conductive material such as pills, UHF tags, which are electrically coupled, may be used to tag the bottle. However, if the bottle contains medicine in liquid form, which is electrically conductive, a magnetically coupled tag, such as HF tag, must be used.

Some early adopters viewed RFID simply as a high-speed technology that replaces barcode. Many had the unrealistic expectation of achieving 100% accurate tag reading and hoped that RFID would be the magic bullet that can increase the operation efficiency without affecting the existing business processes. As these early technology validation projects have shown, missed reads and duplicate reads can still happen even in a carefully designed, engineered, and deployed RFID data collection system. With this realization, most users have abandoned the assumption or requirement that tags be read at 100% accuracy. Instead, these conditions are taken into account in the involved business processes during later-phase pilots and rollouts. Various form of temporal- and spatial-based filtering techniques are used to deal with duplicate reads. For missed reads, the solutions are more finessed. In situations where absolute accuracy is required, exceptions are raised in a business process which triggers a rework subflow so that the error can be properly remedied. For example, in the aforementioned pharmaceutical packing application, every tag must be read correctly in order to comply with the pedigree regulations. The packing process must be designed so that when fewer than expected bottle tags are read in a box, a manual rework subprocess is initiated. In some applications, a business may elect to tolerate a small number of missed reads by inferring the presence of the tagged items using heuristics from other sources, for example, an Advanced Shipping Notice (ASN). An example would be a receiving dock of a consumer goods retailer; the serialized IDs of the merchandise within a shipping container can be inferred by looking up the ASN through their container's ID. When compared with IDs reported by the reader, any missed reads can be identified and rectified.

3.2 Software architecture becomes more encompassing; reuse happens at higher level

As RFID technology matures and gets better understood and integrated into business processes, a similar trend also happens in the software architecture that supports RFID solutions.

During the technology-centric early phase of RFID adoption, the software solution architecture for RFID was correspondingly device-centric. Software and solution providers were mostly concerned with creating device-level middleware to simplify the operation of RFID devices. Some of the major issues were device drivers for readers and printers on different platforms, parsing and converting tag formats (for example, EPC), mapping between physical and logical devices, and simple aggregation and filtering. Solution-level software reuse was very limited if happened at all. Because most of these middleware and

software components were vendor specific, interoperability was also hard to achieve. Using the pharmaceutical packing use case as an example, early solution providers would have a very limited repertoire of low-level software components. For every new deployment of the same use case, whether it is for a different customer or for a different location of an existing customer, a lot of resources are required to re-engineer a solution, sometimes almost completely from scratch, due to the inevitable variations present in the new environment. Clearly, this is not cost-effective and would become a major impediment to the wide adoption of RFID.

Many of these issues are addressed by open standardizations such as EPCglobal. Fig. 3 shows various EPCglobal standards that are applicable to components in the packing process. In the figure, the simplified packing process flow is spread across three layers: at the bottom is the *device layer* where the focus is on controlling the device infrastructure to facilitate reliable data collection. In the example packing process this layer is responsible for turning on the reader for data collection when a case approaches, detecting when the case has left read zone and aggregating tags collected and sending them to next layer. Components in this layer also interact with other types of sensors and actuators in addition to RFID readers. For example, motion detectors, various types of displays, or controls that start/stop conveyer. For example, if tag counts do not match the conveyer is stopped so that the offending case can be subjected to manual inspection later. The tag data from the device layer flow into components in the *filtering and collection* layer. Filtering and collection capabilities are essential features of every RFID based supply chain use case. The *Application Level Events* (ALE) specification which provides for a standard way to declaratively specify filtering and collection requirements greatly enhances the ability to develop reusable components. At the next layer, the *EPC Information Capturing layer,* the aggregated tag information is augmented with business step information and stored in a repository. The *EPC Information Service* specification standardizes formats and interfaces to combine business activity information with RFID tag data and to capture relationships between RFID tagged entities, for example, the unit bottles and the containing case.

The emergence of standards enabled middleware and solution vendors to invest more in the upper levels of the architecture stack and consider RFID and its surrounding business processes in a holistic way. Many vendors have promoted their own specific RFID solution architectures. Although details may vary in these architectures, they share many common design features. In such architecture, RFID is but one component of the overall solution which may consists of multiple inter-connected layers. Thanks to standardization, RFID can now be abstracted away properly as a type of device, alongside with other types of devices such as barcode reader. A network service layer deals with the information exchange within a business and across partners in the framework of EPC network. A process layer consists of the necessary tools and runtimes for orchestrating RFID enabled business processes. An integration layer provides connectors and adapters to external information sources, e.g., trading partners. Finally, the business layer allows RFID enabled processes to be monitored and controller using tools such as dashboard.

With this level of thinking, technologies, such as RFID, are put into their right place as an enabler for business process improvements and innovations that result in the real return on investments. Now, solution providers can start treating higher-level solution artifacts such as process as reusable software assets and leverage their accumulated expertise in process engineering, integration, and optimization to deliver RFID solutions more efficiently.

3.3 New software programming models are gradually adopted

Appropriate software programming environment is needed to process the events generated by the RFID devices. Such an environment includes programming models, tools supporting the programming models, and runtime platforms for executing the artifacts created by the tools. As we have discussed in the preceding subsection, with the maturing of technology and advent of standards, lower-level components that process RFID events have become more reusable. Increasingly, solution providers are looking into reusing higher-level solution artifacts, such as, use cases and processes.

Enterprise application integration methodology and design patterns, such as, Service Oriented Architecture (SOA), have been successfully used in various business process integrations scenarios. In SOA, a reusable piece of software code is packaged as a component, with clearly defined interfaces, which specify the services the component is capable of providing, and references, which indicate the services that this component requires in order to carry out its designed functions. In addition to traditional programming language, such as, Java, C/C++, higher-level business-oriented methods, such as, business rules, decision tables, or process flows, are also used to implement SOA components. Applications or solutions can then be created by connecting individual components via their interfaces and references to form a composite.

Because of the availability of expertise in SOA-related technologies, many early RFID solutions have been built using these existing tools and runtimes from device-level components all the way to processes. As more and more RFID solutions are developed and deployed, the distinction between solution composition at process level and RFID event processing level becomes clear. At process level, reusable assets are abstracted as services, which can be choreographed as procedural steps to form a process. Control flow languages such as BPEL (Andrews et al., 2003) are well suited for solution composition at this level. On the other hand, at the event processing level, reusable assets are more suitably abstracted as agents, which generate and react to events asynchronously. Data flows instead of control flows connect the components to form composites. Although it is possible to implement this data flow semantics using traditional SOA middleware, it is often cumbersome at design time and inefficient at runtime.

For this reason, researchers have promoted event-driven architecture (EDA) as an alternative and complement to SOA to better support the event processing portion of an RFID solution. Event-driven architecture shares many of the same design principles of SOA. A piece of reusable software code is packaged as an agent with an interface, which defines the types of events that it consumes (input ports) and the types of events that it generates (output ports). Hierarchical composition is used to create complex applications or solutions by connecting individual agents to form a data flow diagram. Because this programming model offers a closer mapping to the physical data flow in an RFID event-processing system, it allows solutions to be created more easily, thus lowering the development cost. Because the agents in a data flow exist in their own rights and are not instantiated for each invocation, an application can be executed very efficiently by an EDA runtime to cope with the large volume of RFID event data, further reducing the operating cost of a solution. The reader is referred to (Chen et al, 2008) for an example of an event-driven RFID programming model and run-time system.

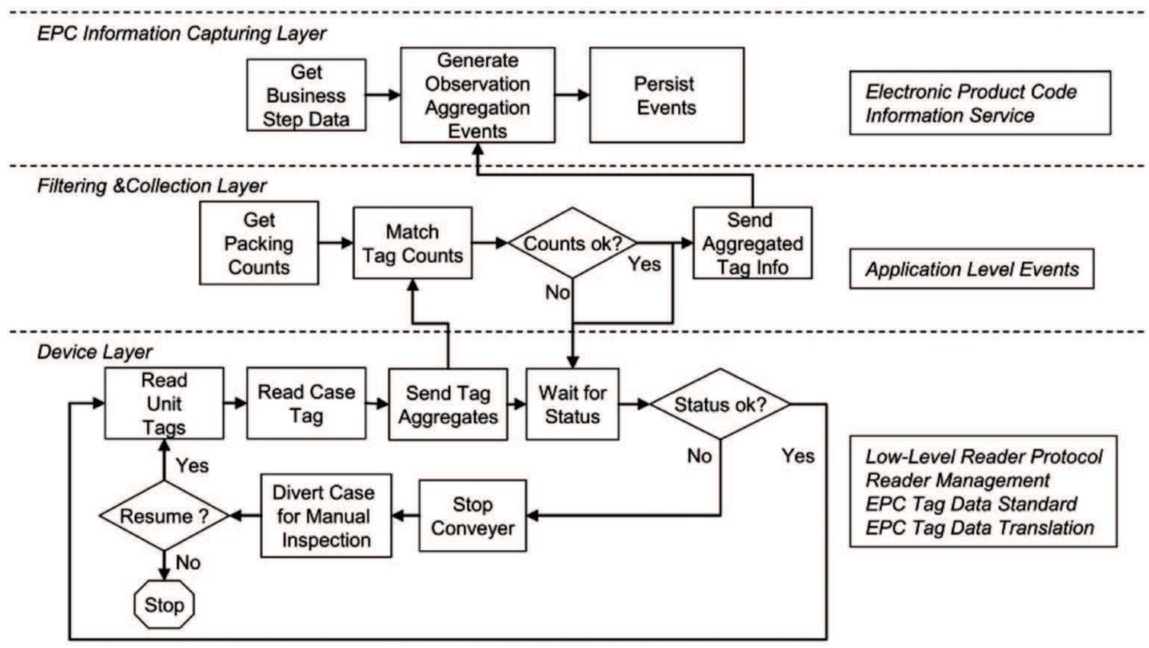

Fig. 3. Mapping of packing process across EPC Network Layers

4. Templatization approaches

There are some factors that could influence how the packing process is carried out in a given setting. The kind of material filled into the unit containers influences the type of tags used to track them. If the unit containers contain liquids, typically, high-frequency (HF) RFID tags are used. Cases, typically, are tagged with ultra-high frequency (UHF) tags. In this setup, shown earlier in Figure 2, two RFID readers are needed. If the unit containers are filled with dry goods, UHF tags can be used and only one reader is needed, and all tags are read at the same time. These are variations that will determine what to set up for the RFID infrastructure. At validation step other factors come into play depending on what type of accuracy we need to attain; in the case of electronic pedigree we must ensure that all individual tags are counted, while in other scenarios we may allow for some degree of error and reconcile with other sources. The process logic should explicitly distinguish unit tags from case tag. Depending on the industry we are addressing we may choose one type of validation over the other. In another variation, when the process is used for packing cases into pallets unit tags are usually filtered out. Furthermore, as this process involves physical movement of goods the underlying device infrastructure may differ: conveyers are normally used for items/units and fork lifts are used for moving pallets. This means the sensing and reporting infrastructure may need to be configured accordingly to support variations at the device layer in Fig. 3.

Let us now observe how a System Integrator that has a team working with a customer to build a packing and verification solution and is using assets from different Independent Software Vendors (ISV) would approach the problem. One of those ISVs has built a verification component that could be configured to integrate with the packing portion of the process. This component is now in its third release and the ISV understands better how

clients are approaching the validation process and integrating it with other components to build solutions. Han the architect of the component has directed his development team to implement a new requirement to support electronic pedigree that originated from the use of the previous versions. Han is planning to generalize the component and expose points of variability or configuration points to parameterize the degree of accuracy that a given component requires before triggering an exception. Back at the System Integrator, Paul, the Business Analyst that has mastered the Packing process and its variations is working with his Solution Architect, Dalia. Dalia understands the RFID domain and the components and processes that could realize that packing process. Based on standards such EPCIS and ALE, Dalia chooses components that are able to interoperate with each other. Dalia assembles an RFID solution that is EPCIS compliant and thus can interoperate with other components they have built in house. Dalia sets the Points of Variability, both pre-deployment and runtime parameters, according to their agreed values and integrates the validation component. A second scenario helps Dalia gain more insight on the component, this time she sets the Point of variability differently but still according to the specification to accommodate the new use case. Dalia has observed that some Points of Variability do not change from use case to use case and could be preset, furthermore she has established reasonable ranges for some parameters, and created some relationships with parameters from other components that conform the solution. Next time the composition of components is deployed fewer parameters will need to be preset and the guidelines and constraints will accelerate the instantiation process. At this point Dalia's experience is captured into what is called a template, a composition of components and possibly other templates that others can use, with the benefit of the encapsulation of default parameters and simplified settings of those that still need to be addressed.

Lily a practitioner at the System Integrator, working with another customer, a manufacturer of shampoos and conditioners, finds the template applicable to her customer's problem and decides to use and apply the template into her solution, the template advises on the RFID infrastructure needed such as the use of two RFID readers, and allows to configure the degree of accuracy required by removing the need of 100% accuracy and lowering the corresponding threshold. This approach requires a much smaller learning curve to use the assets and apply the lessons learned during Dalia's iterations and maturing process while building the template. Sastry, the customer, on the role of solution administrator, is the person that at on-boarding time and during steady state configures a deployed solution to the specific customer needs. He has another opportunity to further change the behavior of the solution; for example, he may select specific external services that fulfil a required function, or other available services to manage the policies that govern the thresholds of scan retries on a per product basis.

4.1 User roles

In the above description we identified the user roles involved in the templatization process following those described in (Fu et al., 2008). We also extend their classification with the Solution Administrator as one that has another opportunity to configure the deployed instance. The Template Creator represented by Dalia, Solution Creator carried on by Lily, the Component Builder impersonated by Han, and the Solution Administrator realized by Sastry.

The Template Creator is usually the Subject Matter Expert, she has enough technical skill to abstract a solution and create related templates such that they can be used multiple times. The Template Creator determines the points of variability and expresses them into the template.
The Solution Creator is able to identify the right template candidate and may be able to perform the configuration tasks needed to instantiate the template. She may perform additional compositions to complete the solution.
A Component Builder has the more in-depth knowledge about each component used in a template. He understands the underlying technology and is able to create new reusable components based on requirements. He creates versatile components with its own customization points that later can be exposed as Points of Variability in the template.
The Solution Administrator belongs to the end-user group; he implements decisions at runtime to customize the solution based on the business needs.

4.2 Template properties

According to the Object Management Group, Inc., Reusable Asset Specification (OMG, 2005) you can look at an asset in 3 dimensions: granularity, variability, and articulation. Larsen (Larsen, 2006) synthesizes them as follow:
Granularity is the spectrum of asset size and shape. Asset may range from fine-grained, meaning they are small in size and purpose, to coarse-grained, providing larger size and purpose and often containing or referring to fine-grained assets.
Variability refers to the asset's degree of customization. This, too, is a spectrum. On one end, the asset is fixed, and on the other, it is widely visible and changeable.
Articulation is the level of completeness of artifacts in an asset that can provide a solution. Some assets have very few artifacts that assist its consumer; whereas others are fully articulated, providing artifacts that include, for instance, requirements and testing validation.
We look at templates through this same asset view. We will explore granularity by looking at the template continuum that we believe are relevant to RFID solutions. Variability will show us the options to compose a solution; it will let us determine the amount of configurability, flexibility on setting parameters, and even potential for late bindings of relevant pieces of the solution. Articulation will show us the degree of realization of a solution.

4.3 Template granularity

Assets range through all levels of the solution building process. At the highest level we can have a solution description that comprises a set of business processes that captures the subject matter expertise around the problem being solved. Our use case in Section 2 around the electronic pedigree is such solution description. The business process may express the possible variations and constraints that allow the process to be valid and yet fulfill distinct solution recipients. An asset may be purely descriptive, documentation only or be a model that captures actors, interactions, and data artifacts on which they operate. An asset may also be accompanied by a set of components, data definitions, business process orchestrations, and user interfaces that could be deployed customized and connected to a given customer scenario. Each of these elements may have a realization in one or more

target environments. For example, a UHF or HF physical configuration depending on the nature of the items (liquid or dry) may fulfill the implementation of the packing infrastructure or at a lower level from a user interface a simple HTML User Interface or a Rich Client Ajax based interface could implement the same interactions between actors for the solution. Each of these elements may be considered an asset on its own, often the distinction between an asset and just an element depends on the value it holds on its own, usually described by its variability, which we will describe later.

Templates as assets behave similarly. At the higher level, a process description may be templatized, suggesting a methodology on how to approach the solution, for example the Track and Trace process could be a template that starts with a process model with its associated artifacts. Next, each particular portion of the process such as packing, or shipping could be a template, the template may compose lower level components into what could translate to a fully functional part of the solution once properly configured and integrated with runnable assets.

If we think how a template comes about, it is the result of multiple iterations of applying an asset or group of assets to a solution that is repeated within some range of change. As we understand better that range we can parameterize those values that affect the behavior that is relevant to the solution. The smaller size assets lend themselves better to incremental modifications, and we will only incorporate other assets at the point that we understand how those interact with the ones that already are part of the template. Figure 4 illustrates this point; the lower layer shows the candidate components and how they can be assembled into a template that supports part of a process.

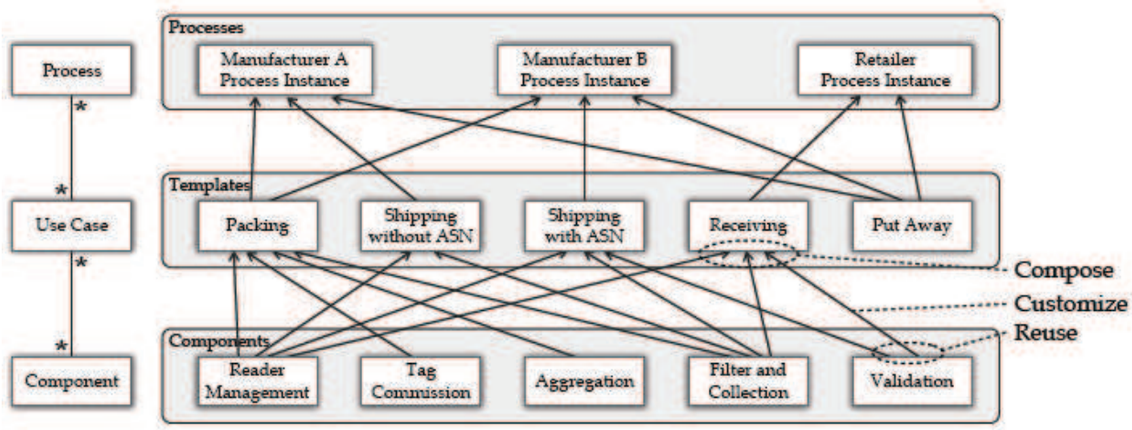

Fig. 4. Solution building using templates

It is often the case that a composition will grow to a point that represents a very particular case, possibly of little interest to anyone else, and with the lack of opportunity to iterate to make any judgment on the amount of combinations that are available it often leads to a set of assets that provides just an example of the integrated components, usually for others a proof point that the technologies employed work and can be used in similar scenarios yet requiring an integration of their own. In a way we could argue that such end to end solutions represent a template yet lacking proper articulation.

4.4 Template variability

An asset is as valuable as its capability to integrate with other assets in the solution. Lack of integration points will limit its useful life. In the past, vendor technologies for COTS (common off the shelf) packages have provided their own mechanisms to integrate; however with the advent of SOA it has become much easier to integrate assets from different sources. As long as the asset exposes a WSDL or REST interface the integration does not require special libraries and just needs the standardized definition of the interface or the resource and the documentation of its use.

There are several implementations of such notions, for example, Apache iy Project (Tuscany) implements the Service Component Architecture or SCA (Besiegel et al., 2007). SCA provides a model for creating composite applications by defining the services in and their relationships with one another. The services can be implemented in any technology. Protocols are pushed out of business logic and are handled through pluggable bindings. Applications can easily adapt to infrastructure changes without recoding since protocols are handled via pluggable bindings and quality of services (transaction, security) are handled declaratively.

As the components are composed into templates the Template Creator builds into them a degree of variability to support the experience he has gained. He then expects the Solution Creator to make use of them through a valid configuration to realize a working solution. Let's consider the more common components in a solution composition: process flows and adapters. A process flow will encompass all related business process constructs such as packing or its associated validation process as described in our example while adapters will comprehend all necessary logic to interact with an external system, an adapter to RFID reader for example. The next subsection examines how variability is built into a template.

4.4.1 Points of variability

To address the customization aspect of a template, the notion of points of variability is introduced. A given artifact may offer customization points or points of variability that identify the options to modify its behavior. We see Variability in two dimensions at Design or Solution Composition time, and post-deployment at runtime, often at customer on boarding time, when the solution is first configured and later tuned up during steady state. The former may require making decisions earlier, usually at pre-deployment time and set parameters one time with little or no opportunity to modify them again, while the latter presents the options at onboarding time to reflect the decisions of the end-user.

For a process flow variability may be introduced at design time on the interfaces that are being offered, some interfaces may be enabled or disabled based on the choice of parameters that may be configured, for example. At runtime configuration parameters that complete the setup of the process flow may take place. (Fu et al., 2008) and their solution template tool focused on design time templates offered the following points of variability for process flows:

- Interface selection – Depending on the features being enabled in the process, interfaces could be conditioned based on property values. The tool supported three types of conditions to govern the point of variability: mandatory, optional and conditional. Some interfaces of the process flow are mandatory and need to be present when the

template is configured. Some interfaces are optional and can be enabled or disabled at configuration time at the discretion of the Solution Creator. Some interfaces will be automatically enabled if the condition associated with the point of variability is met; otherwise, it will be disabled. For more sophisticated conditions rules can be used to evaluate the condition that determines the enablement of the point of variability.

- Properties – An artifact may offer some configuration parameters in the form of name and value pairs, a type may further constrain and provide some level of validation on the possible values. More importantly this value can be extrapolated at the template level to create a global notion that once configured its value propagates to the individual components.

Similarly, for an adapter, points of variability can be defined for the inbound and outbound interfaces, use of properties to set a behavior, etc. One type of variability is the implementation choice, in the case of an adapter, it may be determined by the third party software or service or device we choose to connect to. Depending on the setting of the point of variability a different underlying adapter will be instantiated.

We expect tooling to detect if a variability point is available and present the Solution Creator with options to choose, further more would a mismatch occur, tools offer options to resolve the incompatibility. The idea is to minimize the second guessing by the user and drive the configuration process through completion into a valid configuration where the artifact may operate.

Note that we have used the terms configuration and customization interchangeably throughout this chapter. In practice we refer to configuration when we have a set of parameters that needs to be set and they are strictly defined usually with proper ranges and consistency rules. We refer to customization when a more complex and likely longer cycle takes place to vary the behavior of a component. For example, a validation function may have to be implemented to support a user requirement, or an adapter for a non-standard peripheral may need to be incorporated to complete the solution. Although configuration is preferred over customization the nature of these complex solutions often require a degree of more sophisticated flows, managed by a varieties of policies and constraints that may require additional implementation. One approach that has been used to bridge the gap effectively is with the use of Business Rules as described by Linehan (Linehan, 2005). Rules as another artifact in the SOA composition can be referenced from a component to dictate a point of variability in the behavior of such component.

4.5 Template articulation

In essence, the use of SOA allows for a modeling capability that could be platform independent, as long as the component builders are able to express the components interfaces as service interfaces and provide its configurability through a service. A given component could be realized with various implementations. Templates themselves could also be specified as an SOA assets allowing the notion of sub-templates. Once the template is configured per the user requirements and a platform selected, it can be provisioned as a solution instance in that target platform. The template itself does not pose any tie-in to the underlying technology; the resulting realization of the solution should be standalone and work under the design and runtime tools of that platform. This is also a common use of templates, where the template acts as a starter and the native design tools of the target

platform continue to enrich the solution. Integrated template tooling such as SIMPLE (Laredo et al., 2009) further provide the traceability to the design tools allowing to capture any refinements made and expose them again in a new version of the template for future use.

Other technologies such as Model Driven Business Transformation (Bhattacharya et al., 2007) allow to capture business process models including the business artifacts relevant to the process and the roles that are involved. It provides transformation capabilities to create platform independent state machines for each business artifact and user view, and easily translate to a target platform and map into existing components. Customization opportunities exist along the way; from modifying the base Business Process to extending the compositions at each step of the process. This approach provides for a better articulation of a template as the transformations and customizations ultimately end in an executable solution.

5. Conclusions

RFID has been shown to deliver business benefits in supply chain management such as reducing stock-outs and improving on-shelf availability (Corsten, 2005) (Hargrave, 2006). Such benefits can only be realized by judiciously integrating the technology into related business processes. In this chapter, by using a pharmaceutical product track and trace application as an example and drilling down the manufacturer's product packing process, we examined the evolution of RFID projects and the emergence of solution templates. In essence, templates capture the knowledge and assets applicable to a problem domain and allow for rapid solution realization and deployment. We discussed the attributes of templates to be considered for addressing points of variability and asset reusability that are crucial for increasing ROI and accelerating time to value.

The exploitation of SOA and its integration with EDA allow developers to capture the invariant process logic and defer the decisions about the specific service implementations and device choices to deployment time. The result is that the logical design of a process can be preserved while the underlying components and devices can continue to evolve over time. With solution templates, in most cases, changes in business process or requirements can be easily addressed by 1) changing parameter value, 2) changing the composition flow, 3) adding, deleting or choosing a different implementation of components., and 4) applying business rules None of these requires low-level coding.

The pharmaceutical industry is a good example of an industry segment that has standards applicable to all layers of EPC network application stacks. Such a standard based stack promotes the development of components that cover functional requirements while leaving many non-functional aspects for vendors to compete in implementations and allow for later customizations, thus laying the foundation for template building. We expect the development of industry oriented standards to accelerate and cover many more industries and applications.

The decision whether to reuse an existing asset is a complex one and a number of factors have to come together to tip the balance markedly in favour of reuse. To better facilitate asset reuse decision and process, additional technology enhancements are required in the following areas. One of them is easy-to-use asset repositories for discovering and depositing

assets. Another is an environment to quickly learn various aspects of an asset in question such as usage examples, runtime simulation of asset compositions etc. Yet another factor is the existence of standards reducing asset learning curve and increasing its understanding.

6. References

Andrews, T., Curbera, F., Dholakia, H., Goland, Y., Klein, J., Leymann, F., Liu, K., Roller, D., Smith, D., Thatte, S., Trickovick, I., Weerawarana, S., Business Process Execution Language for Web Services, *Version 1.1, OASIS,* Needham, MA, 2003.

Bhattacharya, K., Caswell, N. S., Kumaran, S., Nigam, A., Wu, F. Y., Artifact-centered operational modeling: lessons from customer engagements, *IBM Systems Journal, v.46 n.4, p.703-721,* October 2007

Beisiegel, M., Blohm, H., Booz, D., Edwards, M., Hurley, O., Ielceanu, S., Miller, A., Karmarkar, A., Malhotra, A., Marino, J., Nally, M., Newcomer, E., Patil, S., Pavlik, G., Raepple, M., Rowley, M., Tam, K., Vorthmann, S., Walker, P., Waterman. L., SCA Service Component Architecture - AssemblyModel Specification, version 1.0. *Technical report, Open Service Oriented Architecture collaboration (OSOA)*, March 2007.

Fu, S., Yang, J., Laredo, J., Huang, Y., Chang, H., Kumaran, S., Chung, JY., (2008) Solution Templates Tool for Enterprise Business Applications Integration, *Proceedings of IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC 2008) 2008*, pages 314-319, June 11-13, 2008, Taipei, Taiwan.

EPCglobal, www.epcglobalinc.org.

Laredo, J., Yang, J., Jeng, JJ., Perez, G., (2009) SIMPLE: Template Based Service Integration, *Proceedings of IEEE International Conference in Web Services, Industry Summit*

Larsen, C. (2006) Model-driven development: Assets and reuse, IBM Systems Journal, Volume 45, Number 3, pp 541-553

Linehan, M. (2005) SOA programming model for implementing Web services, Part 9: Integrating rules with SOA, *IBM DeveloperWorks*, http://www.ibm.com/developerworks/webservices/library/ws-soa-progmodel9/

Object Management Group, Inc., (OMG), 2005, Reusable Asset Specification, Version 2.2, http://www.omg.org/technology/documents/formal/ras.htm.

H. Chen, P. Chou, N. Cohen, S. Duri, C. Jung, "DRIVE: A tool for developing, deploying, and managing distributed sensor and actuator applications," *IBM Systems Journal, Vol 47, No 2,* 2008.

Corsten, D., and Gruen, T. (2005) On shelf availability: an examination of the extent, the causes, and the efforts to address retail out-of-stocks. In Doukidis, G. J. and Vrechopoulos, A. P. (eds.) Consumer Driven Electronic Transformation: Applying New Technologies to Enthuse Consumers and Transform the Supply Chain, pp. 131–150, Springer, Berlin

Hardgrave, B., Waller, M., and Miller, R. (2006) RFID's impact on out of stocks: a sales velocity analysis. Research article ITRI-WP068-0606, Information Technology

Research Institute, Sam M. Walton College of Business, University of Arkansas, June 2006.

Tuscany home page, 2008. http://tuscany.apache.org/.

Sustainable Radio Frequency Identification Solutions

Edited by Cristina Turcu

ISBN 978-953-7619-74-9
Hard cover, 356 pages
Publisher InTech
Published online 01, February, 2010
Published in print edition February, 2010

Radio frequency identification (RFID) is a fascinating, fast developing and multidisciplinary domain with emerging technologies and applications. It is characterized by a variety of research topics, analytical methods, models, protocols, design principles and processing software. With a relatively large range of applications, RFID enjoys extensive investor confidence and is poised for growth. A number of RFID applications proposed or already used in technical and scientific fields are described in this book. Sustainable Radio Frequency Identification Solutions comprises 19 chapters written by RFID experts from all over the world. In investigating RFID solutions experts reveal some of the real-life issues and challenges in implementing RFID.

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen