Expert Systems with Applications 36 (2009) 7581–7593

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Development of an agent-based system for manufacturing control and coordination with ontology and RFID technology

Ruey-Shun Chen [a], Mengru (Arthur) Tu [b,c,*]

[a] Institute of Information Management, China University of Technology, Taipei, Taiwan, ROC
[b] Institute of Information Management, National Chiao Tung University, Hsinchu, Taiwan, ROC
[c] Innovative Supply-Chain Application Division, Identification and Security Technology Center, Industrial Technology Research Institute, Hsinchu, Taiwan, ROC

ARTICLE INFO

Keywords:
Radio frequency identification (RFID)
Just in time (JIT)
Ontology
Multi-agent system

ABSTRACT

Integrating physical objects with the corresponding enterprise applications any time any where is the essential issue for a real-time enterprise. This study proposes a multi-agent system framework called agent-based manufacturing control and coordination (AMCC) system, a agent-based framework using ontology and RFID technology to monitor and control dynamic production flows and also to improve the traceability and visibility of mass customization manufacturing processes. The capabilities offered by multi-agent systems to respond to RFID events in real-time and a broad class of agent design and coordination issue regarding just in time (JIT) and just in sequence (JIS) manufacturing processes are also exploited in this study. To validate the proposed framework, case study of a bicycle manufacturing company is used to demonstrate how the proposed framework can benefit its JIT production. Finally, an example prototype system is implemented to demonstrate the concept of the proposed framework.

1. Introduction

Many manufacturing companies adapt new information system to monitor manufacturing activities and take immediate action to resolve any emergent event that could cause production disruption or customer dissatisfaction (Byrd, Lewis, & Bryan, 2006). In other words, they change their business operations to provide product variety and customization through flexibility and quick responsiveness, and also to remove the data latency, analysis latency, and decision latency as much as possible (Hackathorn, 2003). Therefore, employing mass customization, JIT and lean production, alone with real-time business intelligence enables a firm to achieve that goal and survive in today's hyper competition environment.

For firms manufacturing high priced and highly customized products with large plant facilities, like automobile or bicycle plant, a combining of good product identification technology and loosely coupled manufacturing application architecture plays a key role to enable flexible and agile manufacturing (Kalakota, Stallaert, & Whinston, 1995). However, current manufacturing applications are most developed using client–server computing technologies that use static interfaces tightly coupled to the implementation of functions within the application (Stojanovic, Dahanayake, & Sol, 2004).

Ubiquitous computing can be combined with existing enterprise information system to create loosely coupled distributed systems and invoked to accomplish a complex manufacturing task. In fact, e-commerce solution using ubiquitous computing technology is not anymore a dream (Gershman, 2002). For a next generation enterprise, evolving into a ubiquitous organization becomes an unavoidable direction (Kotorov, 2002). When an enterprise facing technology and business change, it seeks to build new applications upon its existing strengths and assets for competitive advantage. This frequently entails utilizing new technologies and building new applications by coupling existing or so called legacy ones. Even though solutions to help implement ubiquitous computing applications have been proposed (Arregui, Fernstrom, Pacull, Rondeau, & Willamowski, 2003; Floerkemeier & Lampe, 2004; Langheinrich, Mattern, Romer, & Vogt, 2000), they cannot really support the integration requirements in a ubiquitous and real-time environment.

Radio frequency identification (RFID) technology now provides an additional option – data carriers that can not only be read, but also written (Doerr, Gates, & Mutty, 2006). Now, in addition to recording the identity of an object, it is also possible to document its current status and the past and future of the object (Finkenzeller, 2003). Using modern identification techniques, production systems can now be realized which can produce variants of a product, or even different products, down to a batch size of one (Finkenzeller, 2003).

* Corresponding author. Address: Innovative Supply-Chain Application Division, Identification and Security Technology Center, Industrial Technology Research Institute, Hsinchu, Taiwan, ROC. Tel.: +886 3 5915800; fax: +886 3 5826474.
E-mail address: tum.iim95g@nctu.edu.tw (M. (Arthur) Tu).

0957-4174/$ - see front matter © 2008 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2008.09.068

The purpose of this paper is to improve the traceability and visibility of mass customization manufacturing processes. Therefore, by utilizing RFID and ontology we propose a new agent-based framework for tracking and controlling dynamic manufacturing processes so that it can help business implementing its new integration requirements. Issues with RFID usage in ubiquitous computing applications have been widely studied (Floerkemeier & Lampe, 2004). In this research, instead of only using message or service exchanging to perform integration for an enterprise, we use RFID to "hook" the physical objects in an enterprise and the applications that traditionally are not easy to be integrated.

The rest of this paper is organized as follows: Section 2 describes enabling technology. Section 3 describes related problems and scenario analysis. Section 4 outlines the system design of agent-based framework using ontology and RFID. Section 5 describes the actual implementation and evaluation of the proposed system. Section 6 is conclusions.

2. Enabling technology for manufacturing control and coordination

2.1. RFID technology

RFID tags can be categorized into either active or passive types. Active tags are powered by an internal battery and typically can be functioned in a read/write mode (Want, Fishkin, Gujar, & Harrison, 1999). They operate with up to 1MB of memory and have longer reading ranges because of the internal power supply. Passive tags do not rely on an internal power source. Their operating power is obtained from the transceiver device. However, they have shorter reading ranges and require a higher-powered reader than active tags (Harry, Chow, Lee, & Lau, 2006).

RFID supports three types of memory: read-only memory (ROM), read/write (R/W), or writes once/read many (WORM). A ROM tag is similar to a traditional bar code where it comes equipped with a unique identifier after the purchase. R/W tags are more complicated than ROM tags and are more expensive because they can be written in increments and can be erased and reused. Unlike R/W tags, each field of WORM tags can be programmed just once. Information can be changed in the tag only once. All the three RFID types are able to embed context-awareness (Paret, 2005).

An RFID framework for ubiquitous computing applications usually should handle the following items (Romer, Schoch, Mattern, & Dubendorfer, 2003):

- Location – geographic information.
- Neighborhood – physical collaboration.
- Time – current time or timing.
- Linkage of the Physical and Virtual World – a link has to be established between tagged physical objects in the real world and the information system.
- History – Some applications do not only react immediately to tag objects entering and leaving the reading range, but objects are also queried about their history later on.
- Context – Typically the application's action when a tag enters or leaves the reader's range not only depends on the identity of the tag, but also on the context such as the earlier presence or absence of other tags.
- Name and Address – the tag must provide some information how an application can access the virtual counterpart.

2.2. Agent technology

With the introduction of RFID technology to enterprise information system, demand for a new kind of software system to process the continuing large influx of RFID data will begin to rise. Agent technology, in this particular situation, becomes a right candidate to take on the new challenge. An agent is an active object which possesses certain capabilities to perform tasks, and it communicates with other agents based on the organizational structure to cooperate the accomplishment of tasks (Lin, Tan, & Shaw, 1998; Weyns, Parunak, Michel, Holvoet, & Ferber, 2005). To cope with the dynamic changing data generated by RFID system, we consider the agent in this paper as a software entity that continuously monitors the data sources in a global computer network where the information of interest is made available in real-time and when certain signals are detected in the data, the software takes the appropriate action on the user's behalf (Kalakota et al., 1995). Agents collectively functioning as an Agencies thus better suited to help anticipate, adapt or predict in response to changing, real-time data than did most current decision support systems (Ferber, 1999; Odrey & Mejia, 2003).

Based on various designs of agent-based system in the literature (Jeng, Schiefer, & Chang, 2003; Lin et al., 1998) and our system framework, we propose a reactive agent architecture that consists of the following components:

- *Communication Channel:* It handles incoming and outgoing messages for an agent. Messages might be coming from tag readers, web service, other enterprise systems, and other agents.
- *Agent Memory:* Each software agent should have its own memory as well where various business rules/logic and current/past state information would reside.
- *Message Parser:* When an agent receives an incoming message from communication channel, its message parser will analyze its message type and corresponding event lists, delivering them to event processor to perform appropriate business operations.
- *Event Processor:* The event processor interprets the decomposed messages delivered from message parser, consults the agent memory for business rules/logics, and then undertakes the appropriate action based on the message type/event lists and their corresponding business rules/logics.

3. Problem statements and scenario analysis

3.1. Case study

The case study, XMbike (a fictitious name chosen in order to preserve the anonymity of the manufacturer), is a global leading bicycle manufacturing company in Taiwan. XMbike markets middle and high-end bicycles to Asia, Europe, and North America with manufacturing plants across Taiwan, China, and South East Asia. A simplified sample model of the overall manufacturing process from frame tube receipt to packing and shipping is given in Fig. 1. XMbike has adopted the Japanese system of Just-In-Time (JIT) to produce their various bicycle models. In addition, its shop floor control environment is loosely coupled and managed by field operators.

3.2. Problem statements

Since XMbike used paper travelers with bar code labels for tracking and identifying the thousands of frames which are moved around the plant facility each day, traceability was not accurate. Therefore, the current shop floor control and tracking process were inefficient. Without accurate real-time work-in-process information can serious impact the effectiveness of JIT and supply chain planning for XMbike especially if it want to move forward to mass customized production. Analyzing this manufacturing process, potential problems and inefficiencies have been identified and examined below:

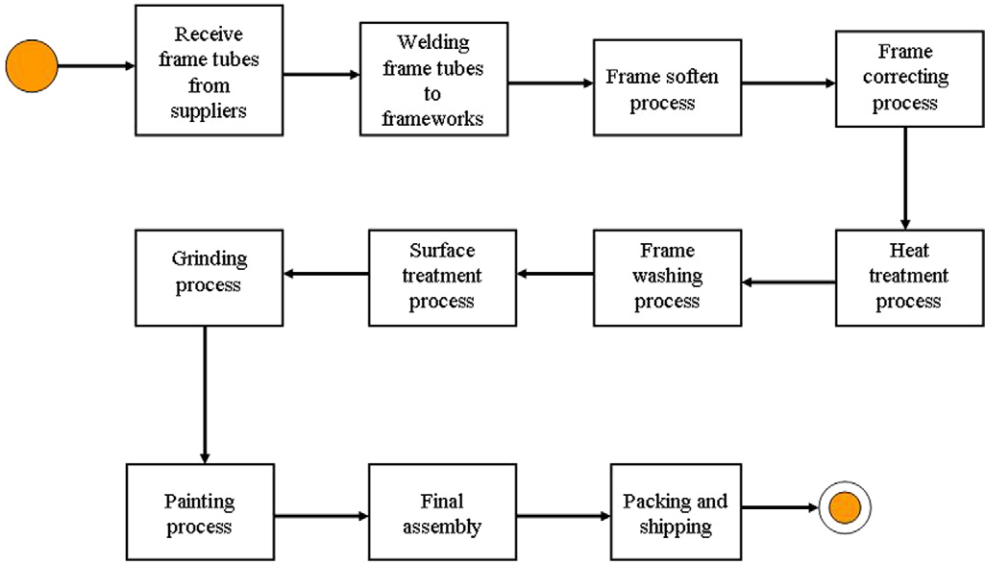

Fig. 1. Overall manufacturing processes of XMbike.

- The barcode system cannot be applied to processes like heat or surface treatment due to the harsh production environment in those processes. Therefore, the firm must employ manual tracking before final assembly stage.
- The Job Card systems of data tracking in XMbike contained only the product build information needed by the production operator at a particular station. The system did not have the capacity for real-time data tracking.
- Time-consuming finding the lost frames due to the large facility space and manual tracking process.
- In some production process, like heat treatment or painting, job card and frame must be separated for a while and then matched back to the exact frame after the process finish. All these actions are must performed by field operator, thus are prone to error.
- Rework process take place from time to time for those process in first floor of the facility. Since frames in those areas are manually tracked, thus rework process sometimes may lead to lost of frames.
- Any missing frames would affect JIT and JIS (Just in sequence) assembly processes in final assembly stage, and it could cost a lot more for mass customization.

3.3. Scenario analysis

Most of XMbike's plants are designed for manufacturing bicycles of limited-choice mass produced model. However, some plants are already manufacturing more high-end model in small lot size. Facing with strong competition from other bike manufactures, XMbike plans to allow customer to select among a variety of model types, frame size, color, and other features. The company estimates that customer can choose from about six million possible variations for such custom-made bike. The process employed to produce such custom-made bicycles requires not only highly trained and skilled workers but also re-engineering of its information system to accommodate such change. Therefore, a mass customization manufacturing strategy naturally lead to the development of a system that rewards attention to details and facilitates operators to achieve 'zero mistakes' in every step of the production process. In light of the strategic goal, XMbike plans to launch a mass customization production line within one of its most advanced plant that currently manufacturing a variety of high-end models with small lot sizes and special orders.

4. System analysis and design of agent-based manufacturing control and coordination (AMCC) system framework

Based on nature of manufacturing process we propose an agent-based manufacturing supply chain coordination framework by utilizing RFID tags as the physical connectors that integrate the bolts and nuts for the physical objects and the enterprise applications and ontology for effective communication between agents.

4.1. Proposed AMCC system analysis

In light of the aforementioned business scenario of XMbike, we intent to design an agent-based manufacturing control and coordination (AMCC) system for this bicycle manufacturer. The RFID system infrastructure in XMbike's plant should include RFID reader positioned at strategic points and RFID tags attached to appropriate position at bicycle frame. In XMbike's case, the RFID Tag is mounted onto one of bicycle's frame's tubes. A RFID tag must survive in harsh manufacturing environment, so a more expensive heat resist passive read/write tag was purchased and then encapsulated by customized designed material to allow the tag to be hook onto the lower part of the frame tube. Local control PC must also be deployed to control RFID readers and provide client side shop floor control service. A suggested multi-agent system deployment layout is illustrated in Fig. 2 and Fig. 3.

4.2. Multi-agent system architecture and agent communication model for AMCC

The goal of this research is to provide a multi-agent framework, which is designed to perform intelligent collaborative supports for JIT (Just in Time) and JIS (Just in Sequence) production strategies in a dynamic manufacturing supply chain within a factory. We define eight types of agent and their agent models are explained below. Their interactions and relationship diagrams are illustrated in Fig. 4. Except for RFID Event Processing Agent (REA) where each instance of REA is resides in a local control PC, the other types of agent is deployed on a centralized server.

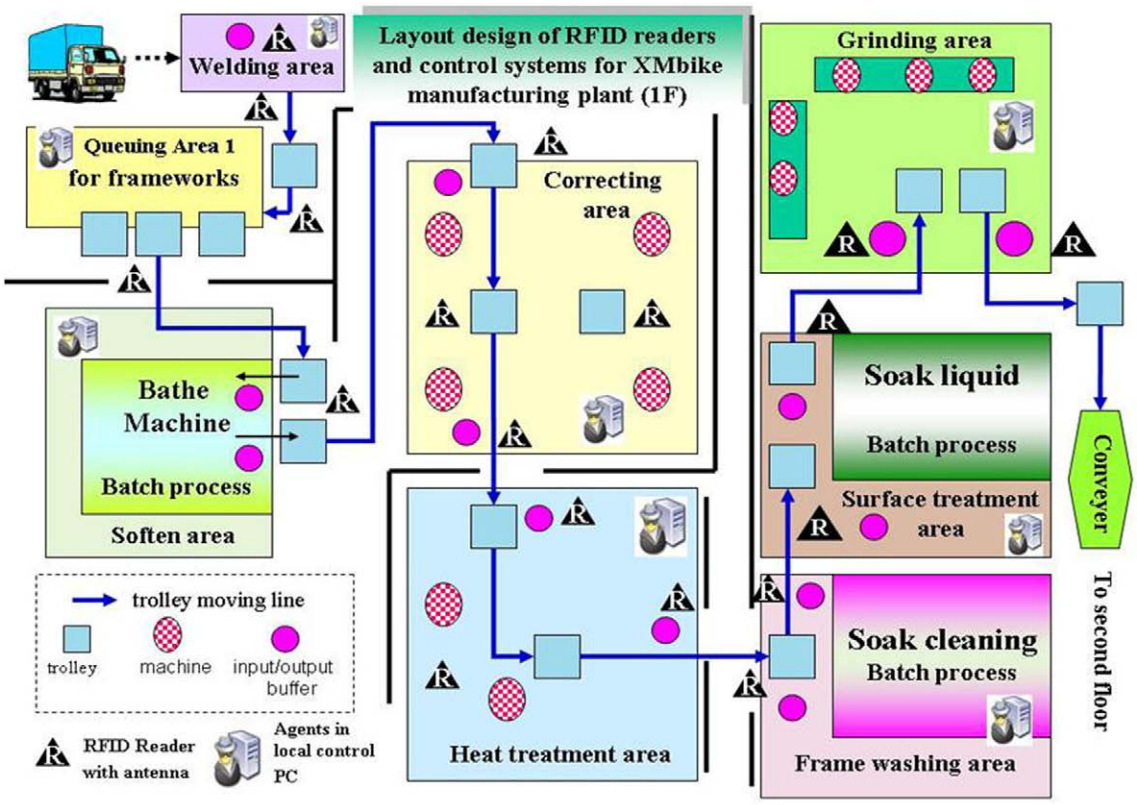

Fig. 2. The proposed AMCC system layout for XM bike – first floor.

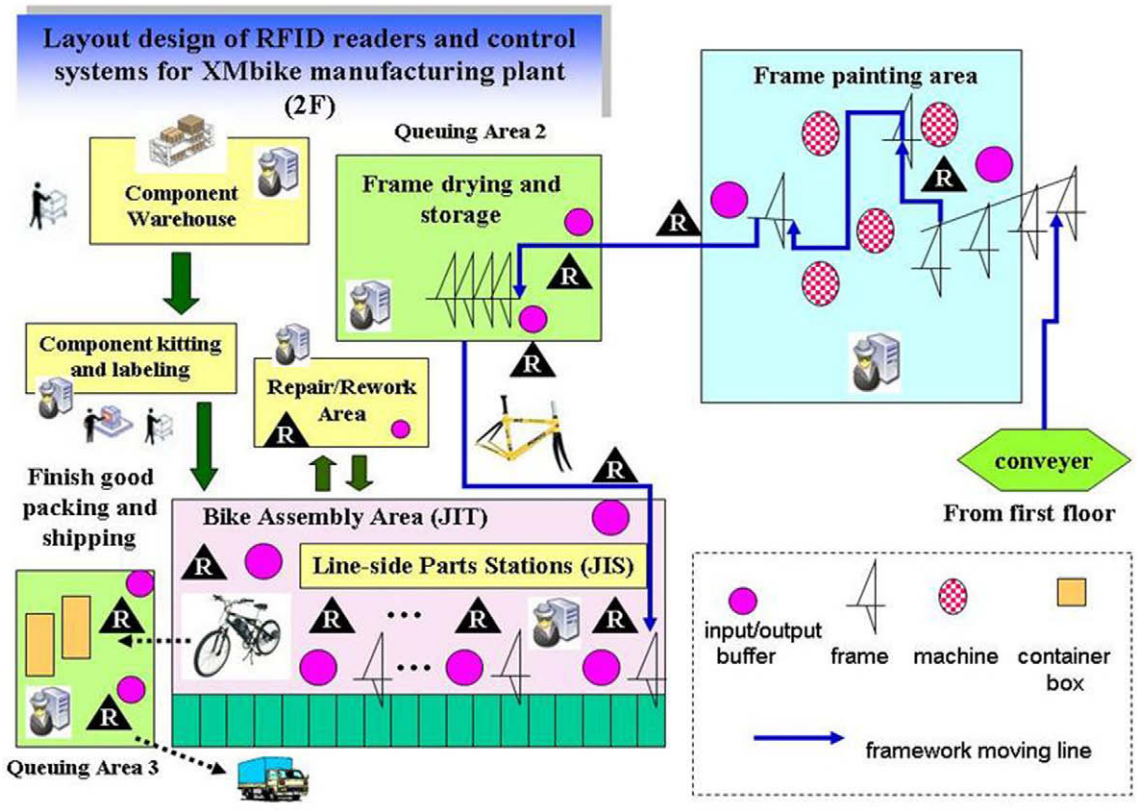

Fig. 3. The proposed AMCC system layout for XM bike – second floor.

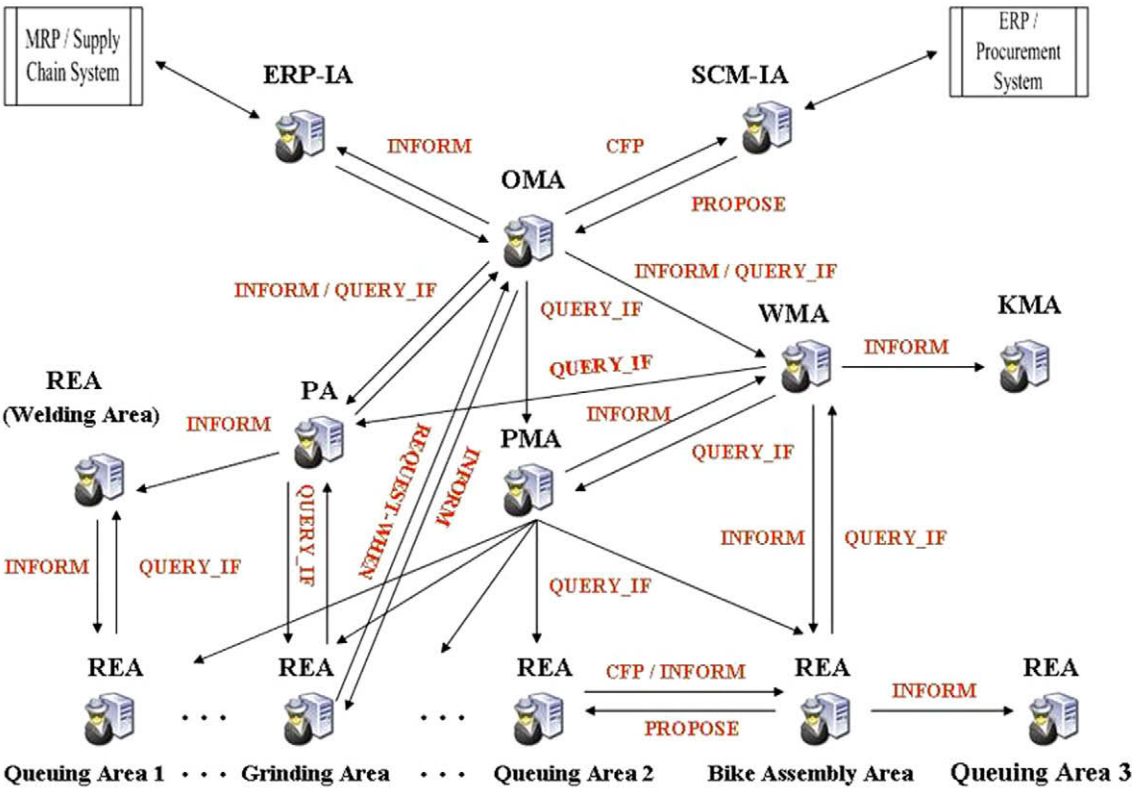

Fig. 4. Agent relationship and interaction diagram for the AMCC system framework.

- Order Management Agent (OMA): An order management agent takes cares of order processing related issue.
- Production Monitoring Agent (PMA): A production monitoring agent constantly monitors the production situation from each RFID Event Processing Agent.
- Warehouse Management Agent (WMA): A warehouse management agent is responsible for calculating available components for bicycle frame in production.
- Kitting Management Agent (KMA): A kitting management agent informs operators to prepare and label manufacturing parts for bicycle frame expected to come to bike assembly area.
- Product Agent (PA): A product agent can generate many instances each corresponds to a RFID tag ID, representing a tagged workpiece, in this case, a bicycle frame in production. Each instance may contains part of RFID tag event data structure like build instruction sector, routing instruction sector, and dynamic event sector, shown in Fig. 6, which we will discuss in later section. Thus, we call the product agent that can initiate new instances a master product agent and those initiated instances of this master agent are called child product agents. The master product agent serves as an interface to other agents or systems that would like to contact with any product agent instances.
- RFID Event Processing Agent (REA): Each RFID Event Processing Agent may represent a manufacturing zone/area or a workstation. This agent is deployed in local control PC. Combing RFID reader, middleware, and event processing logic in one unit makes the REA somewhat like a smart reader or more formally, an intelligent end point that can work on its own even if the centralized server breaks down. Besides, it is easy to configure and deploy for flexible manufacturing environment.
- ERP Interface Agent (ERP-IA): This agent serves as an interface to ERP system.
- SCM Interface Agent (SCM-IA): This agent serves as interface to SCM system.

We use BRIC (block-like representation of interactive components) formalism (Ferber, 1999; Ordey & Mejia, 2003; Weyns et al., 2005) to model above agent's internal state and its interaction with other agents within the agent environment. In this paper, we will show detailed agent modeling using BRIC modeling formalism only for Order Management Agent (OMA) and RFID Event Processing Agent (REA), since their internal structure are more complex than other agent types. We first discuss the OMA in this section and leave the discussion of REA in later section since the REA involves RFID event processing that need to be further explored and analyzed. BRIC model for OMA is illustrated in Fig. 5, while descriptions of place symbols of OMA are given in Table 1. Place symbols are categorized in two types, denoting either a conventional Petri Net place or a input/output communication place; The former representing agent's internal methods and actions while the later representing communication linkage between agents as well as between agents and their environment.

4.3. Agent ontology model for AMCC system

Ontology is a concept borrowed from philosophy and according to the Oxford English Dictionary it is defined as the "science or study of being". In computer and information science, we attribute the ontology to the specification of a conceptualized domain and express it in terms of computer interpretable format, such as the XML. Ontology is used for agent's knowledge sharing and is becoming a crucial element for building a multi-agent system. Only what can be represented using ontologies can be represented in agent's knowledge bases (Obitko, 2002). Recent work by (Chen & Chen, 2008) describes a multi-agent system that applied ontology and

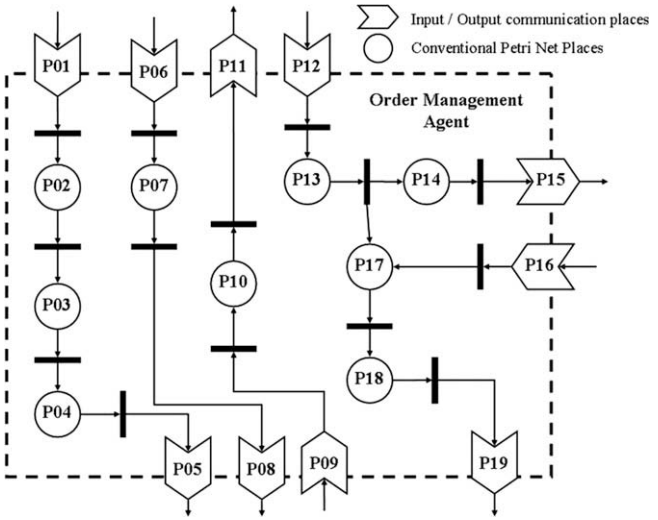

Fig. 5. A BRIC model of an order management agent.

Table 1
Description of places of a BRIC model for an order management agent.

P01	Receiving orders from EPR Interface Agent
P02	Processing orders
P03	Creating product agent based on order information
P04	Initiating product agent with manufacturing and log information
P05	Dispatching product agent to agent environment
P06	Receiving request from SCM Interface Agent (SCM-IA)
P07	Evaluating request from Si M-IA
P08	Sending request to production monitoring agent (PMA)
P09	Receiving feedback information from PMA
P10	Consolidating feedback information from PMA
P11	Sending consolidated feedback to SCM-IA
P12	Receiving request form RFID Event Processing Agent (REA)
P13	Evaluating order priority
P14	Evaluating component availability
P15	Sending request to warehouse management agent (WMA)
P16	Receiving feedback informal ion from WMA
P17	Evaluating and component information from all source
P18	Ranking production priority for workpiece?
P19	Sending ranking results to REA
▬	Transitions represent the start or the end of activities

agent technology to construct virtual observatory is a good example of this trend. According to the FIPA Ontology Service Specification (FIPA, 2008), the ontology is used for agents who want to converse, share a common knowledge for the domain of discourse. JADE (2008) provides the Content Reference Mode (CRM) which is a definition of all information elements in the discourse domain.

Designing the RFID tag event data structure is part of ontology implementation. The design of Mobile Data Carrier (an RFID tag attached to a bicycle frame) data structure is based on XMbike's specific requirements and the overall RFID tag event data structure design is illustrated in Fig. 6. The tag event data structure is divided into ID sector, reserved sector, build instruction sector, routing instruction sector, and dynamic event sector. Expect for the dynamic event sector, the rest of sectors are pre-allocated in tag memory. The information stored in the build, routing, and event sector can be in interpreted by a Dual Tag Data Parser in Tag Handler that is installed in each locally distributed production line control PC.

4.4. Design of RFID tag data structure and its virtual counterpart

Due to the memory capacity constrain of most UHF tag, we may not be able to store that much information into a single tag mem-

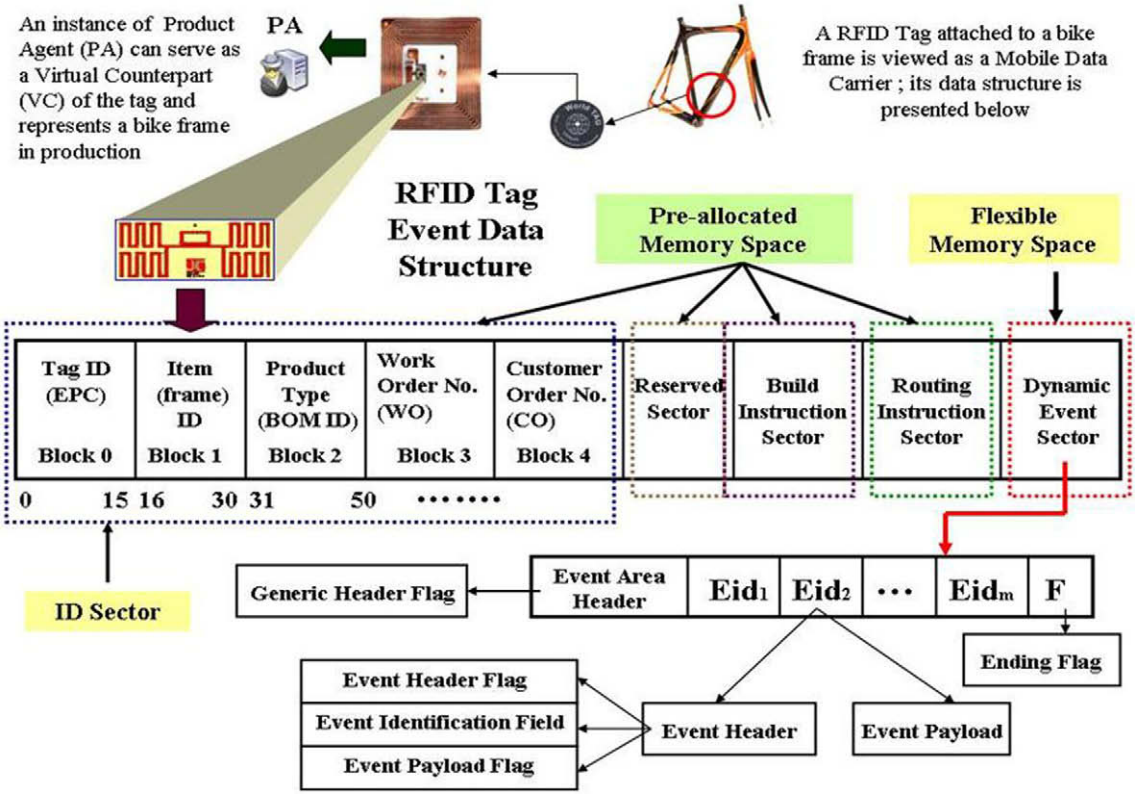

Fig. 6. The RFID tag data structure for the proposed AMCC system framework.

ory at this moment. Thus, we suggest an alternative implementation call Virtual Counterpart (VC) (Romer, Schoch, & Mattern, 2004). In our framework, each instance of Product Agent corresponding to a tag id (which relates to a bicycle frame in production) will manage most of the information of data structure, leaving only ID sector actually reside in the RFID tag memory. The other parts of the data structure of the tag can be stored as XML format and be exchanged between various agents. The example XML format for event sector data structure is shown in Fig. 7. An internal tag ontology parser to parse XML data can be implemented within RFID Event Processing Agent.

4.5. RFID event processing model

For the RFID Event Processing Agent (REA) to manage RFID information well, we proposed a RFID event processing model and later implement it in our AMCC prototype system. RFID event processing model includes RFID tags with specially designed data structure named event lists and software modules to process those events. The memory data structure of an RFID tag mainly comprises three main areas. They are Tag ID, Tag object type and attribute, and event identifier. The details of these data types are explained below:

- Tag ID (tid): Tag id is a unique code of the RFID tag for representing a physical object, and the coding manner can follow an industry standard (e.g. EPC Global or ISO standards) or defined by the company.
- Tag object type and attribute (id_type): An object type and attribute is used for recording a class of the product and associated information of a product combination to facilitate a quick classification of tagged physical objects.
- Event identifier (Eid): Each Eid can contain either a pointer or encoded information to represent actual routing or service instructions, for example, a recipe ID or encoded production recipe for a specific work station. These Eid lists can form a kind of product pedigree which record the history about what events has been taken place for a tagged product until now and can be updated as well as the tagged workpiece travels through each workstation alone the production line.

To cope with large influx of RFID signals, a RFID middleware becomes an indispensable part of any RFID application. The middleware is mainly responsible for managing physical RFID readers and filtering raw RFID data, thus providing a logical interface for application users to physical RFID reader management and low level raw RFID data handling (Chokshi, Thorne, & MaFarlane, 2004). In building REA we not only implement a micro version of RFID middleware in this agent, but also extend functionalities that go beyond a middleware. The REA is made capable of processing RFID events and of interpreting the proposed ontology model. The use of software components and other constitutes in REA are shown in Fig. 8, and their roles and responsibilities are described in detail below:

```
<PRODPEDIGREE>
  <EVENT>
    <EVENTID>MX075P0051V11</EVENTID>
      <PAYLOAD>
        <OPID>OP05</OPID>
        <WORKSTATION>W05</WORKSTATION >
        <CHECKIN>174538</CHECKIN>
        <COMPONENT>
          <COMID>B015S0008</COMID>
          <COMID>B023S0076</COMID>
        </COMPONENT>
        <OPERATOR>XBMAC9546</OPERATOR>
        <CHECKOUT>175535</CHECKOUT>
        <VCODE>V11</VCODE>
      </PAYLOAD>
  </EVENT>
  <EVENT>
  ……
  </EVENT>
</PRODPEDIGREE >
```

Fig. 7. Partial list of event sector information output in XML format.

- *RFID Middleware Control Module:* This module in REA is responsible for managing physical RFID reader and pre-processing of RFID data. It is implemented with some filtering algorithms to handle the redundant or erroneous reads of RFID signals from physical reader. Additionally, a sub-module called **pure tag logging** is also included here. The purpose for adding this sub-module in the RFID Middleware Control Module is to process two types of RFID event separately. Basically, in a manufacturing environment, RFID event can be categorized in two groups. One type deals with complex event processing which usually involves machining processes or assembly operations, the other type handles much simpler event processes like enter/exit an input/output buffer in a workstation or queuing area. For some workstations that are not dealing with machining and assembly processes but with batch chemical processing type of jobs, such as bathe and soak cleaning processes of XMbike shown in Fig. 2, we consider them as simple event type and apply **pure tag logging** to those processes. To simplify the management of multiple RFID readers by REA, we may pre-configure the readers to their appropriate types mentioned above. Finally, some exception handling mechanism for RFID writing operation must also be considered in actual implemented of this module.
- *Tag Handler (Ontology Parser):* The tag handler is responsible for tag read/write process and parsing and analyzing the incoming tag event information. It also facilitates tag handling process based on location specific tag processing logic. It usually requires a RFID Middleware to facilitate its tag read/write processes.
- *Session Controller:* The session controller primary performs data access, data cache, and discovery role in the framework. It provides data cache service for sets of parameter data accepted from a Tag Handler instance and location and/or operation specific information retrieved from Session Data. When an RFID event processing cycle is completed, it synchronizes the processing information with Session Data and releases memory resource in a computer server. This caching and RFID event session cycle management service provided by Session Controller help improve the overall RFID event processing speed and data integrity.
- *Service Dispatcher:* The service dispatcher is responsible for receiving an event processing notification from the tag handler agent, communicating with session controller, and directing an event to corresponding services. It serves as an interface between the framework and external services.
- Event Manager: The event manager is responsible for preparing new events and making a write or clear instruction for the data of the RFID tag through the tag handler. It retrieves new event information from Session Data or receives overwriting event handling instructions from various services and then correctly preparing the information required by the new event and producing a new event code, and requesting the RFID tag handler to write the correct associated event information into the corresponding RFID tag. Additionally, Event Manager might be customized to include some decision making logic in computing optimized production route or be customized to perform product quality checking.

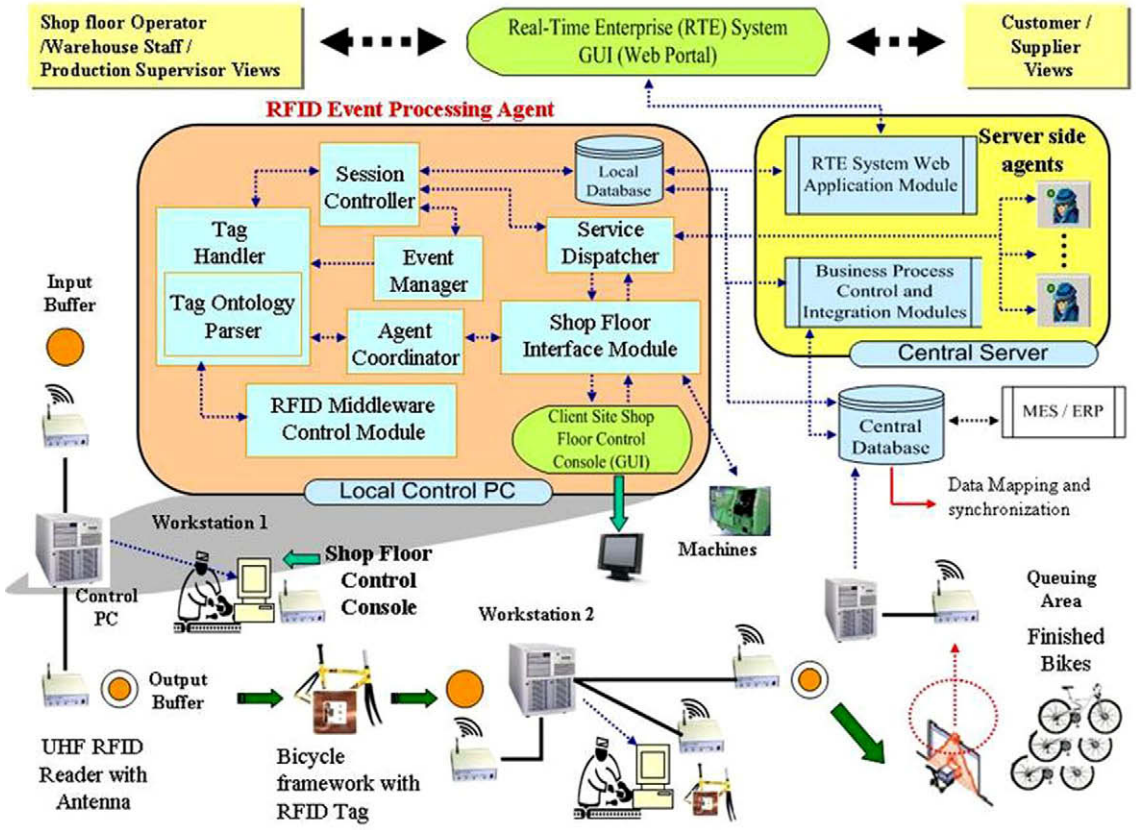

Fig. 8. The proposed AMCC system prototype architecture.

- *Agent Coordinator:* The agent coordinator facilitates interaction among REAs as well as REA to other server side agents. The agent coordination mechanism is devised as a multi-agent decision model in AMCC illustrated in the section of multi-agent decision model.
- *Shop Floor Interface Module:* Serves as a communication channel that relays information from REA to its external environment such as machines or user interface and accepts information from external sources as well.
- *Client Side shop floor control console:* A user interface to facilitate shop floor operation and monitoring.
- *Local Database:* Stored in the local database are two types of dataset described below:
  (a) *Session Data:* In manufacturing environment, this data repository usually resides in line control PC for a workstation or a local production line and stores following types of information:
    - Mapping information for RFID reader/antenna and their covering area/location.
    - Master data regarding each specific workstation in a work area or production line.
    - Recipe related information of local machine/tools in a workstation or production line.
    - System parameters and other data specific to the local workstation or production line.
  (b) *Tag Activity Log:* An operational data store that keeps not only all the tag transaction data as a tagged item goes from one location to another but also some historical information for passing tags. This data repository can be located in central system that monitors all distributed line control PCs. It can also reside in local line control PCs and then periodically sink its information with central system.

The key components involves in RFID event processing and ontology interpretation are Embedded in RFID Middleware Control Programs, Tag Handle, Session Controller, Service Dispatcher, and Event Manager. Their processing steps and information flows are labeled accordingly and represented as a BRIC model of a RFID Event Processing Agent shown in Fig. 9. A detailed presentation of RFID event processing interaction model can also reference (Tu, Jwo, & Kuo, 2006). We map RFID event processing interaction steps to place symbols illustrated in Table 2. Detailed description of each place symbol is presented below:

- P01: A tagged workpiece transmits its signals to RFID reader whenever it enters the reading range of a reader's antenna. RFID reader then passed that signal to the RFID Middleware Control Module. Based on our implemented filtering algorithms, the RFID Middleware Control Module performs data pre-processing by filtering redundant reads and generating a single set of tag data for the Tag Handler.
- P02: After receiving tag data, the Tag Handler immediately parses the data string to obtain tag ID, product type, attribute, and event content. Tag Handler then passes on those parameters to the Session Controller.
- P03: Based on receiving parameter, like product type variable, the Session Controller can retrieve REA's internal parameter information from local the Session Data, for example recipe information of local machine/tools and their corresponding operating procedure.
- P04: The Session Controller then send request, with tag ID and relevant parameters, to Product Agent (PA) to acquire specific production information (like build instruction) of that tagged workpiece.

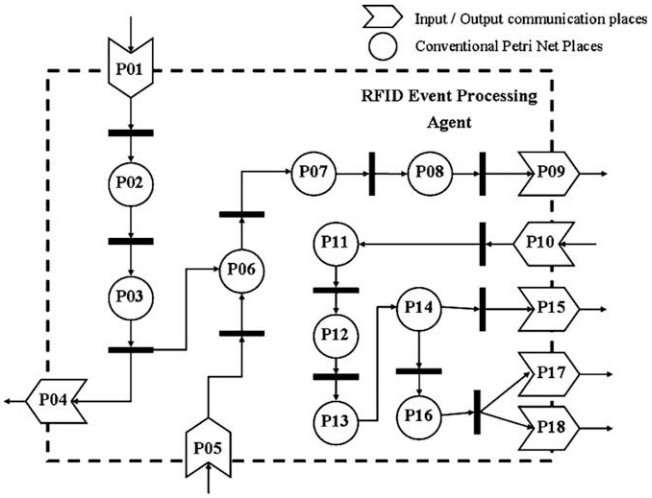

Fig. 9. A BRIC model of a RFID Event Processing Agent.

- P05: The Product Agent (Master) retrieves relevant product information regarding the requested workpiece from its instances (Children) upon request. Product Agent then delivers that information to the Session Controller of the requesting REA. At this juncture, both product information and relevant internal parameters are cached in the Session Controller.
- P06: The Session Controller sends its cached internal parameter and product information to Tag Handler. Tag Handler then evaluates the information to determine whether it is appropriate to process the tagged workpiece by the REA. For example checking whether the tagged workpiece is entering the right station at the right sequence. If there is any error, error handling mechanism will be invoked and operator will be informed immediately.
- P07: If the tagged workpiece is allowed to be processed, the Tag Handler selects appropriate service event for that workpiece and informs the Service Dispatcher.
- P08: The Service Dispatcher prepares service request and operation parameters (event process logic or recipe information) for that service event.
- P09: The Service Dispatcher sends service request and operation parameter to servicing program (a processing machine or operation console) through shop floor interface module and handover the control of workpiece to servicing program.
- P10: The Service Dispatcher receives feedback information from servicing program (completion of servicing task) and then relays the information to the Session Controller.
- P11: The Session Controller sends a completion notice and associated parameters to the Event Manager.
- P12: Based on the parameters of the workpiece, the Event Manager determines the next operation procedure and working area for that workpiece.
- P13: Finally, the Event Manager initiates event update process
- P14: The Event Manager contacts Tag Handler for any information need to be written to physical RFID tag. Since we are using C1G2 UHF RFID tag in our proposed AMCC prototype system, we only put the part of ID sector (as shown in Fig. 6) in the tag due to its limited tag memory space. The rest of RFID tag event data structure is kept in an instance of product agent.
- P15: The Tag Handler writes new event into RFID tag through the RFID Middleware Control Module and RFID reader, usually the information need to be updated to either physical tag or virtual tag like product agent in our case are routing and event information. Thus for our proposed AMCC model this step is skipped.
- P16: The Event Manager informs the Session Controller to prepare writing the associated data registered in its cache memory respectively to the Tag Activity Log in REA's local database and to product agent in central server, thus releasing the corresponding cached event data of the working piece from the memory of Session Controller during this event handling session.
- P17: Session Controller stores workpiece processing information in tag activity log (local database). A server side program will retrieve that logging information from multiple REAs and consolidate them into central database.
- P18: Session Controller sends update information to the corresponding product agent of workpiece residing in central server.

4.6. The multi-agent decision model

As mentioned in Section 3, the Just-In-Time (JIT) production system has been adopted by XMbike so that kanban mechanism is also imbedded in its manufacturing process control. The most important feature of kanban mechanism is that using kanban cards as pull signal the downstream workstation order only what has been consumed (workpieces) from upstream workstation, resulting in the elimination of waste and greater agility of production floor in re-

Table 2
Description of places of a BRIC model for a RFID Event Processing Agent.

P01	Receiving RFID tag data string
P02	Parsing tag string
P03	Loading internal parameter information
P04	Sending request to product Agent (PA)
P05	Receiving information from PA
F06	Evaluating information from internal parameter and product informal ion
P07	Selecting appropriate service event
P08	Preparing service request for service event
P09	Sending service request and parameter information to servicing program (a processing machine or operation console) and handover control to servicing program
P10	Receiving feedback information from servicing program (completion of servicing task)
P11	Inform and relaying information to event manager
PI2	Determining next operation procedure and working area for workpiece
P13	Initiating event update process
P14	Preparing tag update information
P15	Updating information to physical tag
P16	Preparing session update information
PI7	Storing workpiece processing information in tag activity log (local database)
P18	Sending update information to product agent
	Transitions represent the start or the end of activities

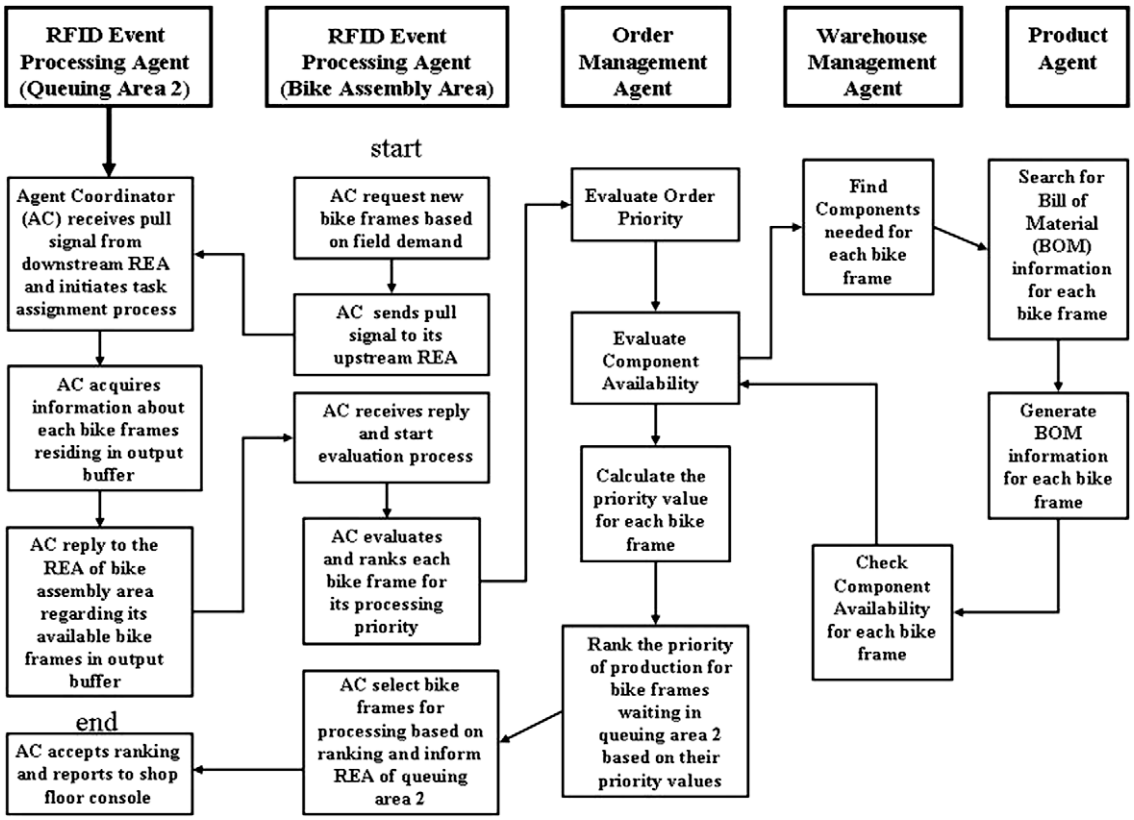

Fig. 10. Multi-agent cooperation and decision processes using pull strategy.

sponse to changing customer demand. In our proposed AMCC, the multi-agent decision model is manifested as a cooperation mechanism among REAs and other agents. This cooperation mechanism in large part simulates the kanban mechanism just described. A simplified scenario of cooperation mechanism among AMCC agents is illustrated in this section. Fig. 10 shows the multi-agent cooperation and decision processes using pull strategy for shop floor control between queuing area 2 and bike assembly area. In XMbike's case the queuing area 2 usually holds a pre-defined optimal quantity of painted fames to serve any request from bike assembly area. After a proportion of painted frames being pulled by assembly area, queuing area 2 would signal to pull new bike frames from frame painting area. Upon receiving the pull signal from downstream stage would trigger the painting processes for the frame painting workstation. To simplified the analysis we only focus on analyzing the dynamics of shop floor control process between queuing area 2 and bike assembly. Five agents are involved in the decision process after the Agent Coordinator of REA at bike assembly area trigger the pull signal. The decision ends at Agent Coordinator or REA at queuing area 2 accepts the processing priority for bike frames residing in its output buffer. The whole process is shown in Fig 10.

5. The prototyping of the proposed AMCC system framework

To validate the proposed framework and prove the applicability and usefulness of the proposed framework to XMbike's operation, we implemented a prototype system, including software components and hardware devices, to demonstrate the aforementioned purposes.

5.1. Development environment

The development environment for the prototype system is described below:

- OS: server: WinXP, client: Linux workstation.
- Database: SQL Server.
- Application program: J2EE technology stack (JSP, JDBC, Java Beans, etc.).
- Application Server: Tomcat.
- RFID Tag: EPCglobal Gen 1 or Gen 2 RFID Tag.
- RFID Reader: AWID UHF RFID Reader.

5.2. System implementation

The prototyping of AMCC presents an integrated information system (Fig. 8) used in shop floor control environment to provide real-time tracking and tracing and adaptive controlling of dynamic mass customization manufacturing process. The agent-based system of this research is build upon Java Agent Development (JADE) platform. JADE architecture enables agent communication through message exchange based on FIPA specifications of agent communication language (ACL). Agents in the JADE environment can work collectively to achieve common objectives by coordination toward a better process. Thus, JADE simplifies the implementation of multi-agent systems (Bellifemine, Caire, Trucco, & Rimassa, 2006). The prototype system is divided into two parts, the real-time enterprise management system on backend central server and the RFID Event Processing Agent (REA) system residing on local control PC.

- *RFID Event Processing Agent (REA):* This part of system implements RFID Middleware Control Module, Tag Handler (including Tag Ontology Parser), Session Controller, Event Manager, Service Dispatcher, Agent Coordinator, Shop Floor Interface Module, and Client Side Shop Floor Control Console. All these components alone with a database are resided in a local control PC and constitute an intelligent end point. A standard REA usually manage more than one set of RFID readers. For example, in a working area both input and output buffers are equipped

with a RFID reader or a reader antenna and if there are any machining or assembly stations, a RFID reader with antenna may be assigned to one or a set of stations, as shown in Fig. 8. The RFID readings from machining or assembly stations would trigger RFID event processing steps as described in Section 4.5 and Fig. 11, whereas the RFID readings from other places mainly deals with simple event processing and the **pure tag logging** will be invoked to perform just tag information logging without further propagating information to Tag Handler and triggering a series of complex RFID event processing logic. **Pure tag logging** usually only records the tag id, workstation id, process id, and the timestamp when tagged workpieces enter/leave a working or queuing area.

- *Real-time Enterprise (RTE) management system:* The backend part of the system consists of server side agents and three sub-modules to take care of specific system operations. The data mapping and synchronization programs are responsible for most data intensive operations like data extraction, consolidation, and synchronization. The other sub-module consists of RTE system web application programs, which support most of front-end, user interface operations of the Web Portal (Fig. 12). It also receives requests from Service Manager Agent and performs certain tasks accordingly. The last sub-module, Process Control Module, updates and maintains manufacturing process and routing information that allows supervisor or manager to access accurate and real-time processing information. As for server side agents, we only implement the product agent at this stage of prototyping and further implementation of other agents proposed by AMCC framework are left for future work.

5.3. System operations

The prototype system simulates the shop floor operation for XMbike manufacturing environment. The facility layout consists of two facility items: queuing area and production control area. When the MES system transforms the job order into working orders (WOs) and notifies our prototype system for the arriving of new working orders, the prototype system then extracts relevant information regarding that WO from MES database and consolidates that information into the central database. A production supervisor can releases the WO, and the RFID tags attached to bicycle frames are written to in this stage with item id and product type, other information are initiated in a product agent instance, such as build and routing instruction. As a frame move into a workstation a series of events in a processing lifecycle are described in Fig. 11. To simulate the JIT pull production process we design the system to allow a shop floor operator to initiate pull command from **client side shop floor control console** of workstation 2. The pull signal will be forward to REA of workstation 1 and the **agent coordinator** of workstation 1 would notify the operator (displayed on control console of workstation 1) to move a finished workpiece to input buffer of its next operation at workstation 2. Fig. 8 shows such scenario. While the manufacturing operation is ongoing, the senior manager can always access the real-time information regarding the where-about of all workpieces as well as current status of workstations through real-time enterprise management system portal as shown in Fig. 12. Our RFID enabled prototype system can virtually eliminate manual barcode scanning and run card (paper travellers) tracking currently used by XMbike.

5.4. Evaluation and benefits

The surveyed company currently uses paper travelers (run cards) with bar code labels for tracking and identifying the thousands of workpieces (bicycle frames) moving around its production facility each day. Tracking and tracing its manufacturing process information involves human intervention that often causes errors and information delay. These problems further affect its process quality control.

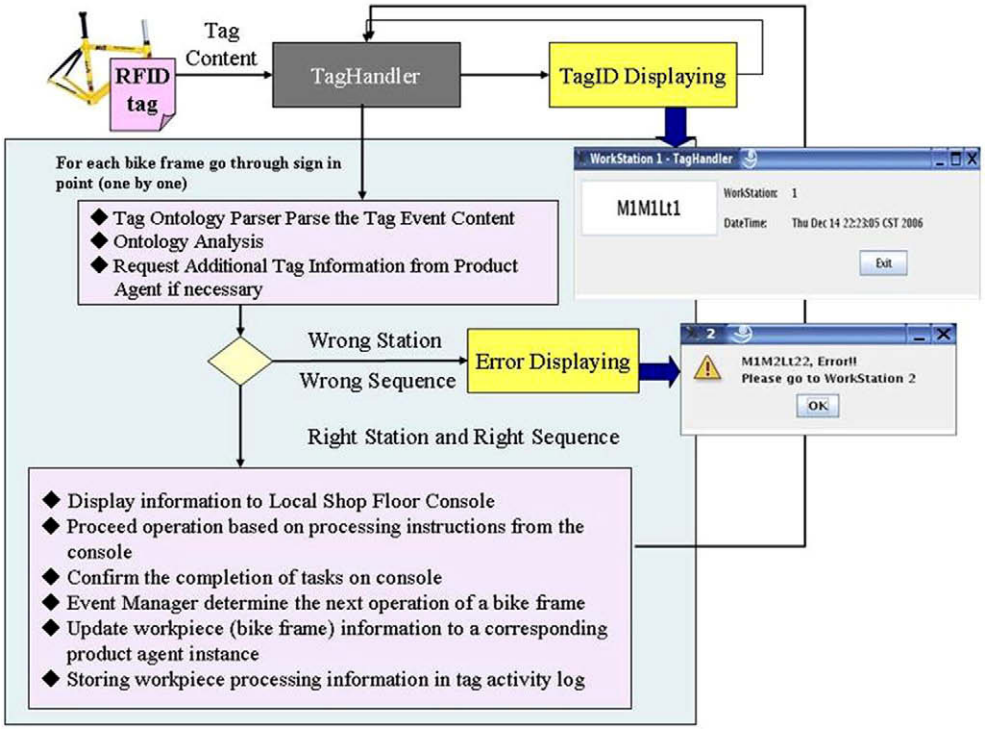

Fig. 11. RFID event processing steps.

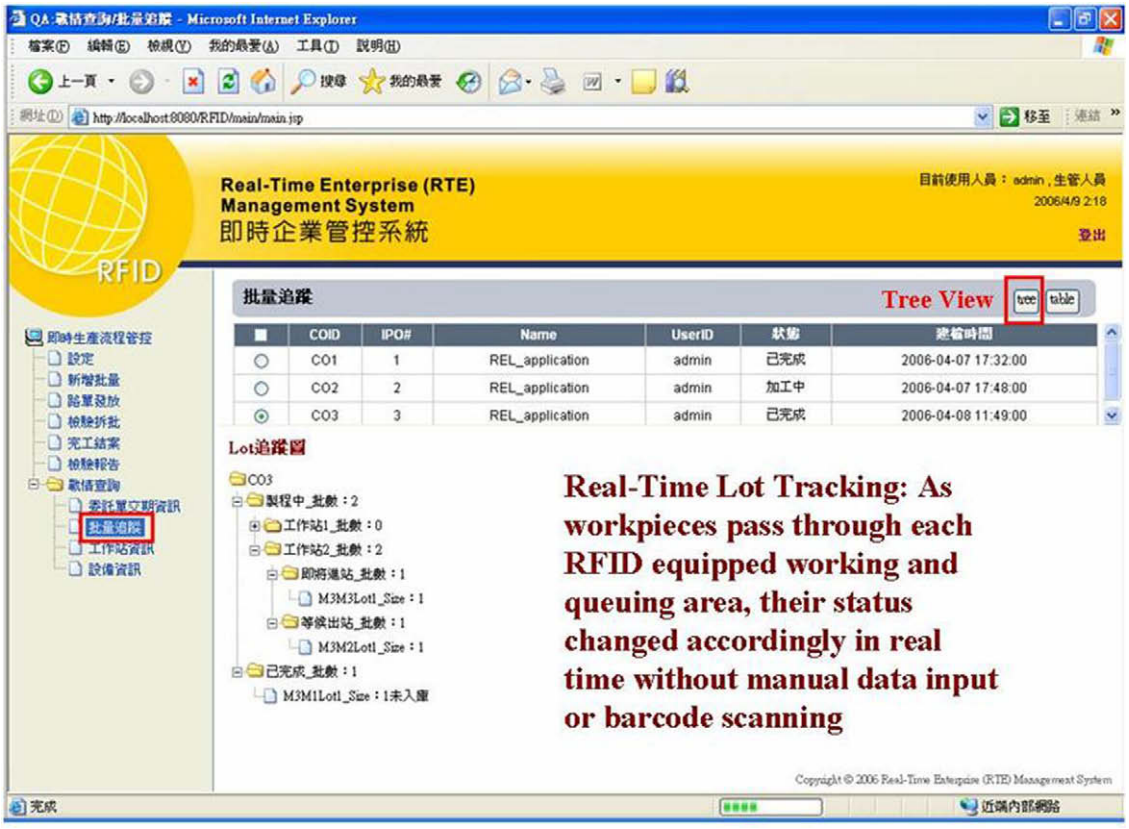

Fig. 12. RTE management system portal.

Table 3
Comparison of manufacturing process tracking between two information systems.

Criteria	Barcode and Run Card (paper travelers) based tracking system	RJTD based multi-agent system
Convenience	Requires line of Tight scan	Automatic scan without line of sight
Efficiency	Cannot support batch reading	Can read multiple tags at once
Accuracy	Susceptible to misreades and human error	Reduced human error and misreads improves data accuracy
Traceability	Limited traceability Some harsh manufacturing process (like baking) make barcode tacking impossible	Allows detailed tracking and tracking of process status, input/outputs, and the time that each processing step was performed
Speed	Process information not in real-time	Real-time process information
Reliability	Barcode are easily dirtied or scraped m Karsh manufacturing environment	RFID tag can survived harsh environment like dirty or high temperature
Automation	Need more human labor to collect and track process data	Replace or reduce human labor in data collection and tracking
Information	Limited process information	Vast amount of detailed process information
Storage (data)	Allows only centralized data storage	Mixed of centralized and decentralized

In our proof of concept pilot research, we found that the overall manufacturing process monitoring and control was improved by the application of RFID technology and our prototype system. Table 3 presents the comparison of information tracking capability between that company's current tracking technology and our proposed one. It seems obvious that RFID technology can significant close the gaps between product flow and information flow. In addition to improved tracking and tracing capability, the prototype system also greatly enhanced the company's quality control of its manufacturing process as our system's real-time validation mechanism help shop floor operators detect quality problems immediately and can resolve these problems before they propagated to the next manufacturing process.

6. Concluding remarks

Manufacturing enterprise transferring into mass customization production and ubiquitous and real-time organization is one of the major trends for enterprise information system. However, to become a real-time and agile enterprise without information technology to help integrating related information and physical objects at different time and different location is quite impossible. In summary, this research makes following contributions to improve the traceability and visibility of mass customization manufacturing processes.

- Proposed a RFID and agent-based manufacturing control and coordination framework for tracking and controlling dynamic manufacturing process flow.
- Based on the framework, we implemented an integrated prototype system to demonstrate the concept.
- The findings of this research reveal that the prototype system can provide both shop floor operator and production supervisor with real-time production process information, helping them to respond to the status of the production line in real-time and make better decisions in handling production events.

- The proposed framework can indeed improve manufacturing process and quality control.

References

Arregui, D., Fernstrom, C., Pacull, F., Rondeau, G., & Willamowski, J. (2003). STITCH: Middleware for ubiquitous applications. In *SOC'2003 – European smart objects conference* (pp. 15–17).

Bellifemine, F., Caire, G., Trucco, T., & Rimassa, G. (2006). *JADE programmer's guide*. <http://jade.tilab.com/doc/programmersguide.-pdf>.

Byrd, T. A., Lewis, B. R., & Bryan, R. W. (2006). The leveraging influence of strategic alignment on IT investment: An empirical examination. *Information & Management, 43*(3), 308–321.

Chen, R., & Chen, D. (2008). Apply ontology and agent technology to construct virtual observatory. *Expert Systems with Applications, 34*, 2019–2028.

Chokshi, N., Thorne, A., & MaFarlane, D. (2004). *Routes for integration auto-ID systems into manufacturing control middleware environments*. White Paper, Auto-ID Center.

Doerr, K. H., Gates, W. R., & Mutty, J. E. (2006). A hybrid approach to the valuation of RFID/MEMS technology applied to ordnance inventory. *International Journal of Production Economics, 103*(2), 726–741.

Ferber, J. (1999). *Multi-agent systems: An introduction to distributed artificial intelligence*. New York: Addison-Wesley.

Finkenzeller, K. (2003). *RFID handbook: Fundamentals and applications in contactless smart cards and identification*. Wiley.

FIPA, The Foundation for Intelligent Physical Agents, <http://www.fipa.org/>, 2008.

Floerkemeier, C., & Lampe, M. (2004). Issues with RFID usage in ubiquitous computing applications. *Lecture Notes in Computer Science, 3001*, 188–193.

Gershman, A. (2002). Ubiquitous commernce-always on, always aware, always proactive. In *Proceedings of the 2002 symposium on applications and the internet (SAINT 02)* (pp. 37–38).

Hackathorn, R. (2003). Minimizing action distance. *The Data Administration Newsletter, 25*.

Harry, K. H., Chow, K. L. C., Lee, W. B., & Lau, K. C. (2006). Design of a RFID case-based resource management system for warehouse operations. *Expert Systems with Applications, 30*(4), 561–576.

JADE, Java Agent Development framework, <http://jade.tilab.com/>, 2008.

Jeng, J. J., Schiefer, J., & Chang, H. (2003). An agent-based architecture for analyzing business processes of real-time enterprises. In *IEEE proceedings enterprise distributed object computing conference* (pp. 86–97).

Kalakota, R., Stallaert, J., & Whinston, A. B. (1995). Implementing real-time supply chain optimization systems. In *Proceedings of the conference on supply chain management*.

Kotorov, R. (2002). Ubiquitous organization: Organizational design for e-CRM. *Business Process Management Journal, 8*(3), 218–232.

Langheinrich, M., Mattern, R., Romer, K., & Vogt, H. (2000). First steps towards an event-based infrastructure for smart things. In *Ubiquitous computing workshop at PACT* (pp. 1–13).

Lin, F. R., Tan, G. W., & Shaw, M. J. (1998). Modeling supply-chain networks by a multi-agent system. In *IEEE proceedings of the thirty-first annual Hawaii international conference on system science* (pp. 5–114).

Obitko, M., & Marik, V. (2002). Ontologies for multi-agent systems in manufacturing domain. In *Proceedings of IEEE 13th international workshop on database and expert systems applications, DEXA'02*.

Odrey, N. G., & Mejia, G. (2003). A re-configurable multi-agent system architecture for error recovery in production systems. *Robotics and Computer Integrated Manufacturing, 19*(1–2), 35–43.

Paret, D. (2005). *RFID and Contactless Smart Card Applications*. Wiley.

Romer, K., Schoch, T., Mattern, F., & Dubendorfer, T. (2003). Smart identification frameworks for ubiquitous computing applications. In *Proceedings of the IEEE international conference on pervasive computing and communications* (pp. 253–262).

Romer, K., Schoch, T., & Mattern, F. (2004). Smart identification framework for ubiquitous computing applications. *Wireless Networks, 10*(6), 689–700.

Stojanovic, Z., Dahanayake, A., & Sol, H. (2004). Modeling and design of service-oriented architecture. *IEEE international conference on systems, man and cybernetics* (pp. 4147–4152).

Tu, A., Jwo, J. S., & Kuo, J. Y. (2006). An enterprise application integration framework using RFID. In *Proceedings of the 36th CIE conference on computers & industrial engineering* (pp. 2750–2760).

Want, R., Fishkin, K., Gujar, A., & Harrison, B. (1999). Bridging physical and virtual worlds with electronic tags. In *Proceedings of ACM CHI '99* (pp. 15–20).

Weyns, D., Parunak, H. V. D., Michel, F., Holvoet, T., & Ferber, J. (2005). Environments for multi agent systems state-of-the-art and research challenges. In: Lecture notes in artificial intelligence (vol. 3477). Berlin, Heidelberg, New York: Springer.