On-the-fly Collaboration for Legacy Business Process Systems in An Open Service Environment

Lin Ye, Biqi Zhu, Chenglong Hu, Liang Zhang
School of Computer Science, Fudan University
Shanghai Key Laboratory of Data Science
Shanghai Institute of Intelligent Electronics & Systems
Shanghai, China
Email: {linye,zhubq17,clhu17,lzhang}@fudan.edu.cn

Hong-Linh Truong
Department of Computer Science
Aalto University
Helsinki, Finland
Email: linh.truong@aalto.fi

***Abstract*—Dynamic, distributed and open business forces enterprises to support various critical requirements, such as, timely reacting to changes, properly reusing business assets and smoothly collaborating with external partners. Existing approaches focus on mechanisms dealing with heterogeneity, but there is a lack of frameworks enabling legacy business processes performing collaboration in an open service environment. This paper proposes the L2L service framework featuring reactive IoT event messaging and coordinator-based collaborating between autonomous enterprises. Along with the emerging of coordinators, L2L empowers on-the-fly business process collaboration with dynamic changes. We present our experiments with a real-world scenario from the shipping industry of China.**

***Index Terms*—cross-enterprise collaboration; open environment; BPM; IoT; legacy systems**

I. Introduction

Nowadays, various business processes, usually represented as a kind of composite web services, are supporting broad sectors of business operations. Business processes and services are shifted to dynamic, distributed, and open environment. Thus, proper cross-enterprises collaborations heavily rely on the capability of uncertainty handling. Empowering legacy enterprise information systems (EIS) with this capability becomes a challenging work because, on one hand, these systems are a kind of assets on which enterprise's routine business relies, and, on the other hand, these EIS have to meet requirements of ever-changing environment. It is imperative to develop a lightweight and non-intrusive framework for EIS to share context and controls among collaboration partners.

To date, IoT technologies have been used to capture status of business entities to be managed within a *single* enterprise in a large-scale system [1–3], where the enterprise manages various workflows/business processes of services to deal with status of business entities. In their collaboration setting, different enterprises also have to coordinate changes through their enterprise services to make sure they meet agreed business constraints among them. However, the current ways [4–6] do not work well with dynamic changes with enterprises captured by modern IoT and context information shared in an open environment, which is characterized by less accessibility, more uncertainty, constant change autonomously, and number of possible interaction partners rises/decreases fast. Usually, available approaches, e.g. [5], rely on ontologies that serve the *unique* semantic foundation for cross-enterprises collaborations. However, this uniqueness assumption hinders them from working in open environments due to the lack of ontology-commitment.

Considering above challenges and the state of the art, we formulate following research questions:

RQ1 SCHEME: the effective approach to support collaborations in open environment
RQ2 FLEXIBILITY: reaction to changes or opportunities
RQ3 USABILITY: mechanism to facilitate long-tailed patches to legacy business processes models.

Through a clear motivation (Section II), we devise a reactive framework named L2L (Section III) that contributes (i) an annotation/event-handling mechanism to recondition legacy business process models on-demand, (ii) incorporating IoT services into processes and smart decision-making based on events triggering functions, and (iii) serverless coordinators for interacting parties (Section IV). We leverage legacy process management systems and validate our work against a real-world case study (Section V). After analyzing related work (Section VI), we conclude the paper.

II. Motivating Example

In achieving the best cost-performance, a bulk shipping company focuses on its core business, and outsources affiliated activities, e.g., the Ship Spare Parts replenishment (SSP), resulting in a cross-enterprise collaboration depicted in Fig. 1. When a Vessel in voyage applies for replenishment of spare parts, the Manager of the shipping company will place an order to an Supplier who then tickets a Logistics company to deliver them to the vessel. The functional requirement is letting the Vessel and a wagon of the Logistics meet at certain wharf on the Vessel's route, and the non-functional requirements are i)delivering parts as early as possible, and ii) saving freight cost as much as possible.

However, the successful delivery *ex ship* is interfered by various asynchronous events (with dotted line), e.g. notifications from harbor (on the left) or traffic jam (on the right), which results various uncertainties. Since nothing shared among

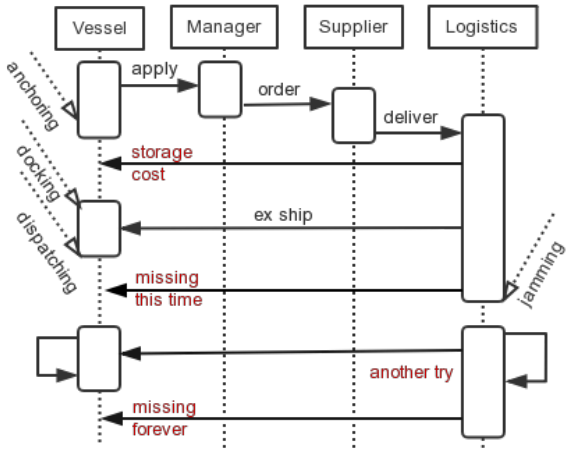

Fig. 1. Ship spare parts replenishment by cross-enterprise collaboration : the successful delivery *ex ship* is interfered by various asynchronous events (with dotted line), which results in various uncertainties (in red).

autonomous participants, it is a challenge to reach balance among the cost, the timing and the uncertainties.

To reach a cost-efficient solution, each collaboration would seek *the most suitable* partners in the market, which is not predictable in advance (RQ1). Once the partnership is formed, we should facilitate the collaboration for legacy EIS of the two partners (RQ3) and meet constraints. Another tricky issue is to be adaptive to changes during the collaboration in order to reach the cost-efficiency of the solution (RQ1 & RQ2). Technically, it is appealing to devise an approach to be (i) reactive, event-based, elastic for dealing changes, (ii) lightweight, non-intrusive solutions for lifting legacy EIS, and (iii) sharing data/context and controls across EIS.

III. Framework

To meet above requirements, we develop a framework L2L[1] as Fig. 2. Many `Enterprises` $\{A, B, \ldots, M\}$ participate collaboration supported by `Cross-Enterprise Coordinators`. When zoomed in, a typical enterprise (e.g. A at top-left corner) manages many `Business Entities` (bottom-left) through business processes in an EIS of the enterprise. Individual enterprise may employ a different workflow/process engine. To facilitate cross-enterprise collaboration among these legacy EIS (in white) by tackling dynamic changes in open environment, we design software services and develop some new components (in grey) that support:

Reacting to changes: `Business Entities` share their IoT data through a `Private Pub/Sub IoT Hub` via `IoT Edge Data Exchanger`. Currently, many IoT enablers have been used for real-world business entities, but here we emphasize a comprehensive solution to make sure that such IoT information can be made available for various other services within the enterprise; `Human Services` are also considered if IoT enablers are not available. L2L provides the capabilities to interface to the EIS through `Event Gateway`, which enables state information exchange between EIS and other services/Business Entities. Thus, within the enterprise, changes and status of entities are timely distributed.

Sharing data/content: L2L enables cross-enterprise collaboration by providing components for sharing suitable business context information in the cloud. `Context Sharing` publishes/fetches events to/from topic-oriented `Public Bulletin` based on enterprise `Policy`. This model provides a controllable mechanism to expose/react-to changes of an enterprise to/from other enterprises based on specific policies for specific collaborations.

Dynamic collaborating: The enabler of collaborations in L2L is the set of `Cross-Enterprise Coordinators`. A coordinator, e.g. the `Coordinator A-B` at the top right corner, takes shared events from the `Public Bulletin` through `Event pub/sub`, which triggers `Serverless functions`, resulting in `Decision Making` based on `Policies`. The decision is feed new events back to the `Public Bulletin` that communicate changes that should make for EIS. It is worth to point out that in L2L (i) coordinators are dynamically generated and localized for specific collaborations and its participants, which releases global terms commitment; and (ii) both serverless functions and policies can be rapidly developed and deployed, therefore highly customized enterprise-to-enterprise collaborations are enabled.

Lightweight, non-intrusive solution: To leverage legacy EIS for reacting to changes and opportunities, business process models are reconditioned by process designers in IFTTT[2] - style annotations and the underlying engine is patched with a specific parser or a new EIS' event handler. This combination makes a tolerable solution to long-tail changes since it equips functional and non-functional capabilities, but for those EIS without them, annotations are totally neglected. Last but not least, local intentions within an enterprise will be checked with collaborations by the `Global Validator` that guarantees the soundness of collaborations across multiple coordinators.

IV. Process Annotation, Context Sharing and Asynchronous Coordination

In realizing the framework L2L, we design components as classes or interfaces as Fig. 3. Due to the space limit, we elaborate the rationale of the framework here but only outline annotation, event sharing, and coordinator design. Technical details are referred to other sources [7, 8].

A. Process Annotation and Activation

In L2L, we use annotation to recondition business process models for new situations, balancing between the usability and maximization of legacy business assets. We choose annotation mechanism because it is extensively supported by various business process modelers, and is tolerable by BPM engines. The non-intrusion is realized by reforming EIS' event handler, or embedding a specific parser into the BPM engine. The two actions can facilitate long-tailed patches for new functionality or constraints to legacy business processes. For example, we

[1]L2L means collaboration between two legacy EIS

[2]https://en.wikipedia.org/wiki/IFTTT

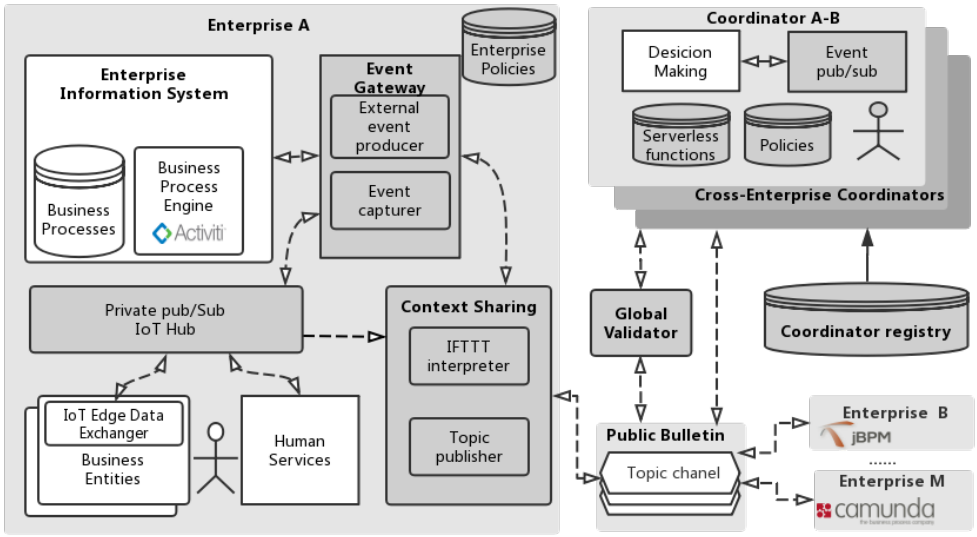

Fig. 2. L2L: a non-intrusive service framework coordinating collaborations of legacy processes from multiple enterprises and reacting to changes.

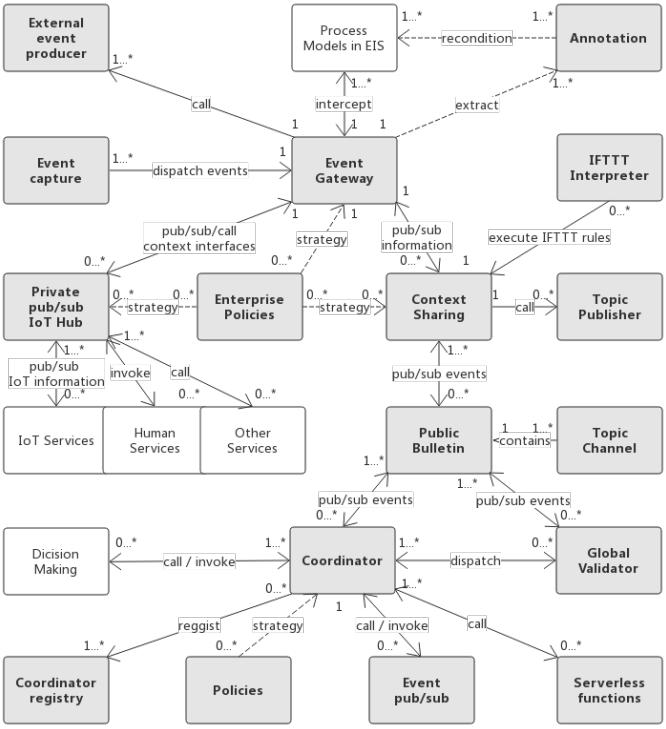

Fig. 3. The concept model of the Framework L2L.

can start from plain shipping functionality, to IoT-enabled, then SSP-enhanced, up to non-functional cold-chain constraints guaranteed with adding annotation and new features.

Process annotation utilized to incorporate new functionality or non-functional constraints to existing business process models – there are three work to do: (i) defining new business variables or constants, e.g. the position of a vessel; (ii) configuring interfaces for new features (functionality and non-functional constraints, e.g. interfaces for the IoT facilities and serverless functions of coordinators; (iii) configuring expressions that trigger functions, e.g. recomputing expecting arriving time, rescheduling rendezvous port, or temperature threshold for turning on air-conditioner, etc. We use the approach of [7] for meeting the three requirements and leverage it with serverless functions and policies. Besides, we develop an IFTTT-style annotation facility to help process designers, rather than relying on IT staffs, to patch business models. The IFTTT-style approach brings us several benefits to reuse business assets. First, it is effective, demonstrated by many applications (e.g. smart home apps) around IoT and user-oriented workflow development. Second, with the help of coordinators, it decouple a collaboration into two separated segments that can be easily reconstructed by coordinators.

Fig. 4 illustrates a case of annotation that utilizes IoT equipment, context sharing methods, enterprise coordinators, etc., to define a new scenario of vessel delay due to the anchoring event. The processing pipeline starts for a plain vessel business process with annotations.

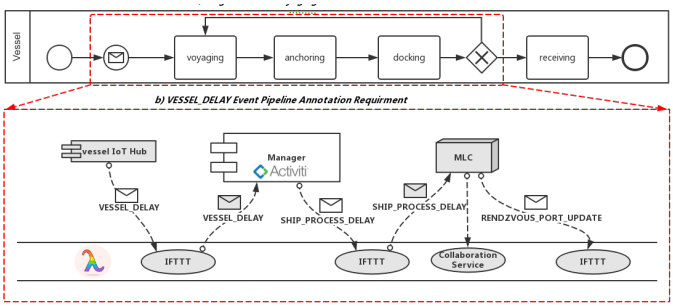

Fig. 4. Annotating Vessel Process with the Capability of Reacting to Vessel Delay due to the Anchoring Event. a) the original vessel voyaging BPMN process, b) annotation requirement added on the original process

Dynamic activation enabled by specified annotations in running BPM systems – the L2L utilizes the Event Gateway which will intercept all the events from the legacy system as [9] by Event capture, and re-dispatch the event with the External event producer based on Enterprise Policies. The L2L utilizes Event Gateway to intercept BPMN process activity events, and integrate IoT-Hub, Context Sharing, etc.

Thanks to the non-intrusive nature of our annotation and dynamic activation mechanism, business assets (process models) are kept, meanwhile, new functionality and non-functional constraints (decorated process models) can be deployed and executed, without any intervention of IT staffs, resulting in an on-the-fly solution to collaboration.

B. IoT Context Sharing Across Networks

Through the activation of annotation, contextual information is ready to share. The question is what to share and how to share it across enterprises. For example, in the SSP problem, the rendezvous port can be dynamically adjusted to meet economic benefits only-if both the vessel and wagon can report their docking information or traffic condition dynamically. In this case, we could have *IoTLocation* from GPS position and *IoTVelocity* describing current average velocity. Whenever the wagon gets stuck during running, it will trigger an event to the business process, firing some business rules in the BPM classical model.

In order to share the context information across enterprise processes, L2L framework first extracts the IoT information and services to a local IoT Hub. The event gateway publishes or subscribes the IoT information from the local IoT hub, then extracts required data to the Context Sharing. With the IFTTT interpreter developed in context sharing, the IFTTT annotations will be executed to make new features injected into and reorganized by current EIS.

C. Asynchronous Cross-Enterprise Coordinator

Changes affecting multiple enterprises usually should be coordinated. Here we focus on events that need to trigger functions for coordinating multiple processes at the same time. There is a set of coordinators. Each of them takes charge in the collaboration of two participants through events. That makes the constraints propagate through the EIS. All coordinators should be registered to the Coordinator registry even they can be freely created or destroyed.

Fig. 5 illustrated business models (in each participant's EIS) and coordinators (in red) to facilitate collaboration across enterprises. Events that trigger them making progress are denoted as messages in the BPMN form.

V. Prototype and Experiments

We have implemented the L2L prototype[3]. In the following we outline some main components:

EIS-BPM Engine: Legacy EIS is based on Activiti[4], which is widely used in realistic BPM practices nowadays.

Event Gateway: We implement the event gateway based on the EventDispatcher interface, provided by Activiti.We implement the IFTTT-style rule engine by combining the annotation interpreter with the IoT Hub, coordinator interfaces, and Context Sharing.

IoT Hub: We use AWS IoT to implement IoT Hub that deals with asynchronous messages (e.g. delay in docking or traffic

[3]https://github.com/i-qiqi/L2L

[4]https://www.activiti.org/

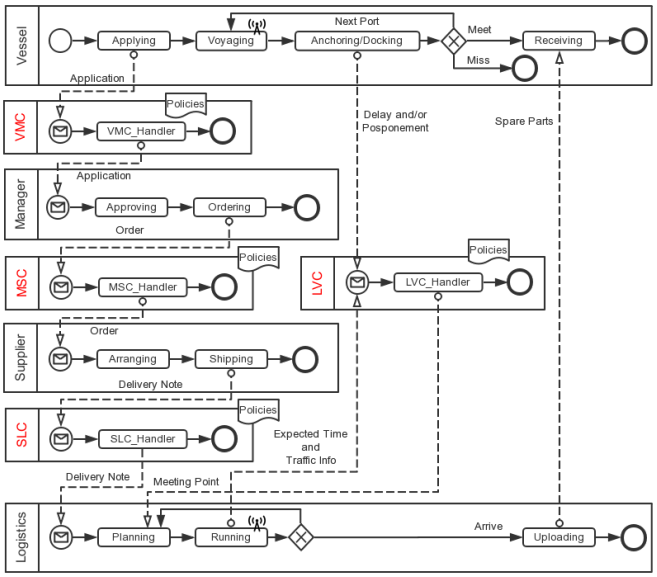

Fig. 5. Coordination and integration of processes from different domains by using coordinators (in red) in the SSP problem

conditions) and event streams (e.g. position and velocity of moving objects). All IoT devices are modeled as IoT shadows, while an IoT service task in Activiti maintains a certain IoT shadow, subscribes certain message, e.g. /wagon/#, in AWS IoT Cloud, and receive all expected data related to the device.

Coordinators: AWS Lambdais used to implement serverless-based coordinators. A user or the Activiti reports information to AWS Lambda via REST service calls, which activates certain functions, written in Python, to perform decision-making according to specific policies. All messages connecting Activiti and Lambda functions are in JSON. Once a function received the message, three agents will be activated:

- *initiator,* gets the feedback among (i) the decision is included in the response part, (ii) the client is attempting to access a resource that does not exist, and (iii) a dependent service is throwing errors.
- *coordinator,* realizes a specific decision-making;
- *collaborator,* posts a request with the decision to the REST interface of the public bulletin.

The initiator and REST interface are implemented as CoordinatorName-handlers task in Activiti.

Context Sharing: This is achievable by using functions to analyze and extract internal IoT data and publish the extracted data to the public bulletin, which has different topics of shared information subscribed by coordinators.

Fig. 6 is a screen-shot from our prototype for SSP problem.

We tested our prototype along the three RQs in Sec. I, against the SSP problem and other scenarios. We found that L2L is a very promising scheme to capture new opportunities as new coordinators emerge. Based on dynamic events captured by IoT, and propagated through L2L, extra value can achieved by taking advantage of change. As illustrated in Fig. 6, based on information from IoT on both the vessel and the wagon, the coordinator LVC (re-)schedules the rendezvous port back and forth dynamically, in responding